[데이터스토리] 데이터로 보는 개봉 영화 (2) 박스오피스

0. 라이브러리 업그레이드 및 한글 사용 설정

0.1. Matplotlib, seaborn, pandas

In [1]:

```
# Step 1. Matplotlib 등 업그레이드
!pip install matplotlib -U
!pip install seaborn -U
!pip install pandas -U
!pip install openpyxl -U

# Step 2. 한글 설치 및 사용 설정
!sudo apt-get install -y fonts-nanum
!sudo fc-cache -fv
!rm ~/.cache/matplotlib -rf

# Step 3. 셀 실행 후 런타임 재시작
```

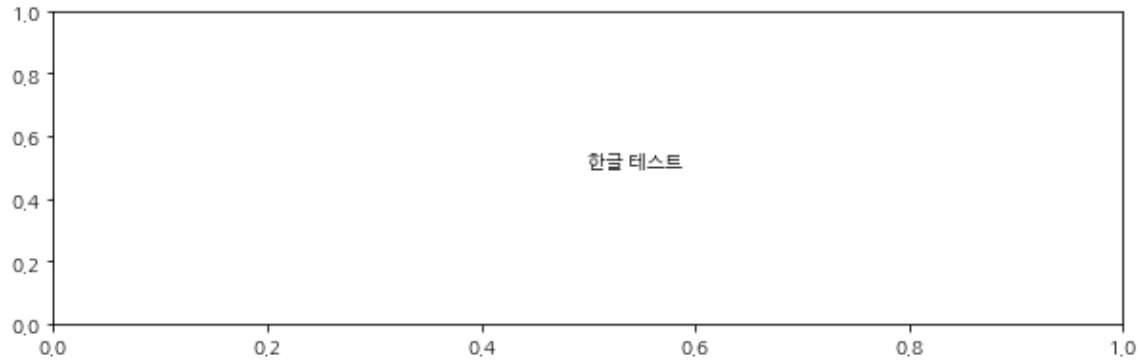
```
Requirement already satisfied: matplotlib in /usr/local/lib/python3.7/dist-package
s (3.2.2)
Collecting matplotlib
  Downloading matplotlib-3.4.3-cp37-cp37m-manylinux1_x86_64.whl (10.3 MB)
     |████████████████████████████████| 10.3 MB 5.3 MB/s
Requirement already satisfied: pillow>=6.2.0 in /usr/local/lib/python3.7/dist-pack
ages (from matplotlib) (7.1.2)
Requirement already satisfied: python-dateutil>=2.7 in /usr/local/lib/python3.7/di
st-packages (from matplotlib) (2.8.2)
Requirement already satisfied: numpy>=1.16 in /usr/local/lib/python3.7/dist-packag
es (from matplotlib) (1.19.5)
Requirement already satisfied: pyparsing>=2.2.1 in /usr/local/lib/python3.7/dist-p
ackages (from matplotlib) (2.4.7)
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.7/dist-
packages (from matplotlib) (1.3.2)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.7/dist-packa
ges (from matplotlib) (0.10.0)
Requirement already satisfied: six in /usr/local/lib/python3.7/dist-packages (from
cycler>=0.10->matplotlib) (1.15.0)
Installing collected packages: matplotlib
  Attempting uninstall: matplotlib
    Found existing installation: matplotlib 3.2.2
    Uninstalling matplotlib-3.2.2:
      Successfully uninstalled matplotlib-3.2.2
ERROR: pip's dependency resolver does not currently take into account all the pack
ages that are installed. This behaviour is the source of the following dependency
 conflicts.
albumentations 0.1.12 requires imgaug<0.2.7,>=0.2.5, but you have imgaug 0.2.9 whi
ch is incompatible.
Successfully installed matplotlib-3.4.3
```

```
Requirement already satisfied: seaborn in /usr/local/lib/python3.7/dist-packages
(0.11.2)
Requirement already satisfied: numpy>=1.15 in /usr/local/lib/python3.7/dist-packag
es (from seaborn) (1.19.5)
Requirement already satisfied: scipy>=1.0 in /usr/local/lib/python3.7/dist-package
s (from seaborn) (1.4.1)
Requirement already satisfied: matplotlib>=2.2 in /usr/local/lib/python3.7/dist-pa
ckages (from seaborn) (3.4.3)
Requirement already satisfied: pandas>=0.23 in /usr/local/lib/python3.7/dist-packa
ges (from seaborn) (1.1.5)
Requirement already satisfied: python-dateutil>=2.7 in /usr/local/lib/python3.7/di
st-packages (from matplotlib>=2.2->seaborn) (2.8.2)
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.7/dist-
packages (from matplotlib>=2.2->seaborn) (1.3.2)
Requirement already satisfied: pillow>=6.2.0 in /usr/local/lib/python3.7/dist-pack
ages (from matplotlib>=2.2->seaborn) (7.1.2)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.7/dist-packa
ges (from matplotlib>=2.2->seaborn) (0.10.0)
Requirement already satisfied: pyparsing>=2.2.1 in /usr/local/lib/python3.7/dist-p
ackages (from matplotlib>=2.2->seaborn) (2.4.7)
Requirement already satisfied: six in /usr/local/lib/python3.7/dist-packages (from
cycler>=0.10->matplotlib>=2.2->seaborn) (1.15.0)
Requirement already satisfied: pytz>=2017.2 in /usr/local/lib/python3.7/dist-packa
ges (from pandas>=0.23->seaborn) (2018.9)
Requirement already satisfied: pandas in /usr/local/lib/python3.7/dist-packages
(1.1.5)
Collecting pandas
  Downloading pandas-1.3.4-cp37-cp37m-manylinux_2_17_x86_64.manylinux2014_x86_64.w
hl (11.3 MB)
     |████████████████████████████████| 11.3 MB 5.2 MB/s
Requirement already satisfied: python-dateutil>=2.7.3 in /usr/local/lib/python3.7/
dist-packages (from pandas) (2.8.2)
Requirement already satisfied: numpy>=1.17.3 in /usr/local/lib/python3.7/dist-pack
ages (from pandas) (1.19.5)
Requirement already satisfied: pytz>=2017.3 in /usr/local/lib/python3.7/dist-packa
ges (from pandas) (2018.9)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.7/dist-packages
(from python-dateutil>=2.7.3->pandas) (1.15.0)
Installing collected packages: pandas
  Attempting uninstall: pandas
    Found existing installation: pandas 1.1.5
    Uninstalling pandas-1.1.5:
      Successfully uninstalled pandas-1.1.5
ERROR: pip's dependency resolver does not currently take into account all the pack
ages that are installed. This behaviour is the source of the following dependency
 conflicts.
google-colab 1.0.0 requires pandas~=1.1.0; python_version >= "3.0", but you have p
andas 1.3.4 which is incompatible.
Successfully installed pandas-1.3.4
```

```
Requirement already satisfied: openpyxl in /usr/local/lib/python3.7/dist-packages
(2.5.9)
Collecting openpyxl
  Downloading openpyxl-3.0.9-py2.py3-none-any.whl (242 kB)
     |████████████████████████████████| 242 kB 5.3 MB/s
Requirement already satisfied: et-xmlfile in /usr/local/lib/python3.7/dist-package
s (from openpyxl) (1.1.0)
Installing collected packages: openpyxl
  Attempting uninstall: openpyxl
    Found existing installation: openpyxl 2.5.9
    Uninstalling openpyxl-2.5.9:
      Successfully uninstalled openpyxl-2.5.9
Successfully installed openpyxl-3.0.9
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following NEW packages will be installed:
  fonts-nanum
0 upgraded, 1 newly installed, 0 to remove and 37 not upgraded.
Need to get 9,604 kB of archives.
After this operation, 29.5 MB of additional disk space will be used.
Get:1 http://archive.ubuntu.com/ubuntu bionic/universe amd64 fonts-nanum all 20170
925-1 [9,604 kB]
Fetched 9,604 kB in 1s (6,878 kB/s)
debconf: unable to initialize frontend: Dialog
debconf: (No usable dialog-like program is installed, so the dialog based frontend
cannot be used. at /usr/share/perl5/Debconf/FrontEnd/Dialog.pm line 76, <> line
1.)
debconf: falling back to frontend: Readline
debconf: unable to initialize frontend: Readline
debconf: (This frontend requires a controlling tty.)
debconf: falling back to frontend: Teletype
dpkg-preconfigure: unable to re-open stdin:
Selecting previously unselected package fonts-nanum.
(Reading database ... 155219 files and directories currently installed.)
Preparing to unpack .../fonts-nanum_20170925-1_all.deb ...
Unpacking fonts-nanum (20170925-1) ...
Setting up fonts-nanum (20170925-1) ...
Processing triggers for fontconfig (2.12.6-0ubuntu2) ...
/usr/share/fonts: caching, new cache contents: 0 fonts, 1 dirs
/usr/share/fonts/truetype: caching, new cache contents: 0 fonts, 3 dirs
/usr/share/fonts/truetype/humor-sans: caching, new cache contents: 1 fonts, 0 dirs
/usr/share/fonts/truetype/liberation: caching, new cache contents: 16 fonts, 0 dir
s
/usr/share/fonts/truetype/nanum: caching, new cache contents: 10 fonts, 0 dirs
/usr/local/share/fonts: caching, new cache contents: 0 fonts, 0 dirs
/root/.local/share/fonts: skipping, no such directory
/root/.fonts: skipping, no such directory
/var/cache/fontconfig: cleaning cache directory
/root/.cache/fontconfig: not cleaning non-existent cache directory
/root/.fontconfig: not cleaning non-existent cache directory
fc-cache: succeeded
```

In [1]:

```
# Step 4. 한글 글꼴 설정
import matplotlib.pyplot as plt

plt.rcParams['font.family']=['NanumGothic', 'sans-serif']
plt.rcParams['axes.unicode_minus'] = False

# 한글 설정 확인
%matplotlib inline

fig, ax = plt.subplots(figsize=(10,3))
ax.text(0.5, 0.5, "한글 테스트")
plt.show()
```



1. 개발 환경 설정

1.1. 기본 라이브러리

In [2]:

```
import numpy as np
import pandas as pd
import seaborn as sns

import requests
from copy import deepcopy

sns.set_context("talk")
sns.set_style("whitegrid")

pd.options.display.max_columns=50

# seaborn 설정에 의해 파괴되는 한글 설정을 재설정
plt.rcParams['font.family']=['NanumGothic', 'sans-serif']
plt.rcParams['axes.unicode_minus'] = False
```

2. 데이터 확보

2.1. 국가별, 장르별 데이터

- **데이터스토리 (1)에 사용한 데이터**

In [3]:

```
# 영화 목록
url_movielist = "https://jehyunlee.github.io/about/src/2021_datastory_movie/df_movielist.pkl"

# # 국가별 데이터
url_nations = "https://jehyunlee.github.io/about/src/2021_datastory_movie/df_nations.pkl"
url_nationsY = "https://jehyunlee.github.io/about/src/2021_datastory_movie/df_nationsY.pkl"

# # 장르별 데이터
url_genres = "https://jehyunlee.github.io/about/src/2021_datastory_movie/df_genres.pkl"
url_genresY = "https://jehyunlee.github.io/about/src/2021_datastory_movie/df_genresY.pkl"

# 데이터 불러오기
df_movielist = pd.read_pickle(url_movielist)
df_nations = pd.read_pickle(url_nations)
df_nationsY = pd.read_pickle(url_nationsY)
df_genres = pd.read_pickle(url_genres)
df_genresY = pd.read_pickle(url_genresY)
```

2.2. 역대 박스오피스 데이터

In [4]:

```
# 박스오피스 데이터: KOBIS 데이터를 받아 header 일부 수정
url_bo = "https://jehyunlee.github.io/about/src/2021_datastory_movie/KOBIS_boxoffice.xlsx"

# 데이터 불러오기
df_bo = pd.read_excel(url_bo, engine="openpyxl")
df_bo.head(3)
```

Out[4]:

	순번	영화명	감독	제작사	수입사	배급사	개봉일	영화유형	영화형태	국적	전국 스크린수	전국 매출액	전국 관객수	서울 매출
0	1	명량	김한민	(주)빅스톤픽쳐스	NaN	(주)씨제이이엔엠	2014-07-30	개봉영화	장편	한국	1587	1.357484e+11	17613682	3.312123e+
1	2	극한직업	이병헌	(주)어바웃필름,영화사그림주식회사,(주)씨제이이엔엠	NaN	(주)씨제이이엔엠	2019-01-23	개봉영화	장편	한국	1978	1.396480e+11	16264944	3.185866e+
2	3	신과함께-죄와벌	김용화	리얼라이즈픽쳐스(주),(주)덱스터스튜디오	NaN	롯데쇼핑㈜롯데엔터테인먼트	2017-12-20	개봉영화	장편	한국	1912	1.156987e+11	14410754	2.753083e+

In [5]:

```
# 결측치 제거
df_bo = df_bo.loc[df_bo["개봉일"] != 'NaT']
df_bo["개봉일"] = df_bo["개봉일"].astype(str)
df_bo["openYear"] = df_bo["개봉일"].apply(lambda x: x.split("-")[0])
df_bo = df_bo.loc[df_bo["openYear"] != 'NaT']
df_bo["openYear"] = df_bo["openYear"].astype(int)
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:6: SettingWithCopyWar
ning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stabl
e/user_guide/indexing.html#returning-a-view-versus-a-copy
```

In [6]:

```
# 전국 관객수 = 0, 서울 관객수 = 0 영화들. 이상치로 간주하여 제거.
list_bo_zeros = df_bo.loc[df_bo["서울 관객수"] == 0].loc[df_bo["전국 관객수"] == 0].query("1971 <= openYear <=2020")[["영화명", "국적", "openYear"]].sort_values("openYear").apply(lambda x: f"{x[0]} ({x[1]}, {x[2]})", axis=1)
print(list_bo_zeros.tolist())
print(df_bo.shape)
df_bo = df_bo.loc[list(set(df_bo.index) - set(list_bo_zeros.index))]
print(df_bo.shape)
```

```
['마술의 검 (미국, 1972)', '아랑신 (중국, 1973)', '별 3형제 (한국, 1977)', '북치는 여자 (한국, 1987)', '오 내사랑 임꺽정 (한국, 1991)', '우주전사 불의 사나이 (한국, 1991)', '우주에서 온 사나이 (홍콩, 1991)', '용궁에서 온 거북이와 이무기 (한국, 1991)', '오늘같이 좋은 날 (한국, 1991)', '전갈군단 (한국, 1991)', '어허 어이 어이 가리 (한국, 1991)', '스캐너스2 (nan, 1991)', '최가손우2 (nan, 1991)', '캠퍼스 연애 소동 (nan, 1991)', '뽀식이와 꼬마특공단 (한국, 1991)', '패배자 (한국, 1991)', '장미빛 여자 (한국, 1991)', '드래곤볼 (nan, 1991)', '하나님이 어디 있어요 (한국, 1991)', '남장여인 (프랑스, 1991)', '라스트 워리어 (미국, 1992)', '쉘부르의 우산 (프랑스, 1992)', '총각파티 (미국, 1992)', '하이힐 (스페인, 1992)', '천국의 계단 (한국, 1992)', '챔프의 분노 (한국, 1992)', '정열의 카르멘 (스페인, 1992)', '나 이제 너를 잊으리 (한국, 1992)', '역사는 이렇게 시작됐다 (한국, 1992)', '옥유단 : 봉성부인 (홍콩, 1992)', '계부 3 (미국, 1992)', '우리는 그들을 잊으려 했다 (nan, 1992)', '우리는그들을잊으려한다 (nan, 1992)', '겨울연가 (한국, 1992)', '혈락제 (홍콩, 1992)', '세나의 신혼일기 (한국, 1992)', '넝쿨속의 히야신스 (한국, 1992)', '불행한 아이의 행복 (한국, 1992)', '사랑의 이름으로 (한국, 1992)', '네이키드 트루 (미국, 1992)', '터커 (미국, 1992)', '퐁네프의 연인들 (프랑스, 1992)', '돌아온 이탈자 (미국, 1992)', '하얀전쟁 (한국, 1992)', '토요일 밤의 남자 (미국, 1992)', '달은...해가 꾸는 꿈 (한국, 1992)', '크리스마스 인 코네티컷 (미국, 1992)', '워락 2 : 아마게돈 (미국, 1993)', '포이틱 저스티스 (미국, 1993)', '팝콘 (미국, 1993)', '잭 더 베어 (미국, 1993)', '방세옥 2 : 대도무문 (홍콩, 1993)', '화엄경 (한국, 1993)', '내가 성에 관해 알고 있는 몇가지 이야기들 (한국, 1993)', '나이트 아이 2 (미국, 1993)', '불꽃 슛 통키 (한국, 1993)', '블랙 캣 2 (홍콩, 1993)', '영웅들의 날개짓 (한국, 1993)', '파워킹 (한국, 1995)', '아름다운 게임 (한국, 1995)', '이도백화 (한국, 1995)', '배꼽버스 (한국, 1995)', '현대 고혹자 (홍콩, 1996)', '불안은 영혼을 잠식한다 (독일, 1997)', '프리처스와이프 (미국, 1997)', '휘파람부는 여자 (한국, 1998)', '인피니티 (미국, 1998)', '포데이즈 인 셉템버 (브라질, 1999)', '옥토퍼스 (미국, 2000)', '낮은 목소리 3 : 숨결 (한국, 2000)', '프레디타 (멕시코, 2000)', '섹슈얼 이노센스 (미국, 2000)', '이유없는 반항 (한국, 2001)', '게임오버 (한국, 2001)', '골뱅이@실명제 (한국, 2001)', '부킹쏘나타 (한국, 2001)', '사보타지 2 (미국, 2001)', '가면학원 (일본, 2001)', '천사의 시 (한국, 2001)', '캅링크 (한국, 2001)', '어페어 오브 더 넥클리스 (미국, 2002)', '타운 앤 컨트리 (미국, 2002)', '블리트 (프랑스, 2002)', '하이힐 크라임 (미국, 2002)', '토스카 (프랑스, 2002)', '하드 캐쉬 (미국, 2002)', '미스터 바티뇰 (프랑스, 2002)', '미쉬카 (프랑스, 2002)', '아름다운 것들 (프랑스, 2002)', '해피 엑시던트 (미국, 2003)', '핫 칙 (미국, 2003)', '주글래 살래 (한국, 2003)', '동거남녀 (홍콩, 2003)', '프린세스 블레이드 (일본, 2003)', '생령 (일본, 2003)', '헤드 오버 힐스 (미국, 2003)', '신이 버린 특공대 (미국, 2003)', '벤자민 프로젝트 (미국, 2003)', '천방지축 (홍콩, 2003)', '터크 에버래스팅 (미국, 2003)', '언세드 (미국, 2003)', '런투유(RUN 2 U) (한국, 2003)', '메이 (미국, 2003)', '다케시즈 (일본, 2006)', '스네이크 온 어 플레인 (미국, 2006)', '트루 저스티스 (미국, 2011)', '클래쉬 (베트남, 2011)', '만화방 (한국, 2012)']
(23382, 19)
(23274, 19)
```

In [7]:

```
# 박스오피스 데이터 저장
!mkdir data
df_bo.to_pickle("./data/df_bo.pkl")
```

In [8]:

```
# 1971-2020년 데이터 선별
df_bo = df_bo.query("1971 <= openYear <= 2020")
print(df_bo.shape)
```

(23271, 19)

2.2.1. 장르 매칭

In [9]:

```
# 영화목록 데이터
df_movie_bo = df_movielist[["movieCd", "movieNm", "openDt", "repNationNm"]]
df_movie_bo["movieNm"] = df_movie_bo["movieNm"].apply(lambda x: x.replace(" ", ""))
df_movie_bo["openDt"] = df_movie_bo["openDt"].astype(str)
df_movie_bo["openDt"] = df_movie_bo["openDt"].apply(lambda x: f"{x[:4]}-{x[4:6]}-{x[6:]}")
df_movie_bo["영화명_개봉일_대표국적"] = df_movie_bo[["movieNm", "openDt", "repNationNm"]].apply(
lambda x: f"{x[0]}_{x[1]}_{x[2]}", axis=1)

df_movie_bo.head(3)
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:3: SettingWithCopyWar
ning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stabl
e/user_guide/indexing.html#returning-a-view-versus-a-copy
  This is separate from the ipykernel package so we can avoid doing imports until
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:4: SettingWithCopyWar
ning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stabl
e/user_guide/indexing.html#returning-a-view-versus-a-copy
  after removing the cwd from sys.path.
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:5: SettingWithCopyWar
ning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stabl
e/user_guide/indexing.html#returning-a-view-versus-a-copy
  """
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:6: SettingWithCopyWar
ning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stabl
e/user_guide/indexing.html#returning-a-view-versus-a-copy
```

Out[9]:

	movieCd	movieNm	openDt	repNationNm	영화명_개봉일_대표국적
10	20139221	그래비티	2013-10-17	미국	그래비티_2013-10-17_미국
23	20201002	조제	2020-12-10	한국	조제_2020-12-10_한국
27	20010264	아멜리에	2001-10-19	프랑스	아멜리에_2001-10-19_프랑스

In [10]:

```
# 박스오피스 데이터
df_bo_genres = deepcopy(df_bo)

# 영화명 띄어쓰기 제거
df_bo_genres["영화명"] = df_bo_genres["영화명"].str.replace(' ', '')

# 장르 결합 key
df_bo_genres["영화명_개봉일_대표국적"] = df_bo_genres[["영화명", "개봉일", "국적"]].apply(lambda x: f"{x[0]}_{x[1]}_{x[2]}", axis=1)
df_bo_genres.head(3)
```

Out[10]:

	순번	영화명	감독	제작사	수입사	배급사	개봉일	영화유형	영화형태	국적	전국스크린수	전국 매출액	전국 관객수	서울 매출
0	1	명량	김한민	(주)빅스톤픽쳐스	NaN	(주)씨제이이엔엠	2014-07-30	개봉영화	장편	한국	1587	1.357484e+11	17613682	3.312123e+
1	2	극한직업	이병헌	(주)어바웃필름,영화사해그림주식회사,(주)씨제이이엔엠	NaN	(주)씨제이이엔엠	2019-01-23	개봉영화	장편	한국	1978	1.396480e+11	16264944	3.185866e+
2	3	신과함께-죄와벌	김용화	리얼라이즈픽쳐스(주),(주)덱스터스튜디오	NaN	롯데쇼핑㈜롯데엔터테인먼트	2017-12-20	개봉영화	장편	한국	1912	1.156987e+11	14410754	2.753083e+

In [11]:

```
# 데이터 병합 전 Key 확인
list(set(df_movie_bo["영화명_개봉일_대표국적"]) - set(df_bo_genres["영화명_개봉일_대표국적"]))
```

Out[11]:

```
['낮은목소리3:숨결_2000-03-18_한국',
 '천사의시_2001-01-01_한국',
 '포데이즈인셉템버_1999-11-08_브라질',
 '남장여인_1991-01-19_프랑스',
 '북치는여자_1987-07-23_한국',
 '맥베스_2018-10-03_영국',
 '기브잇올_2004-10-14_일본',
 '드림메이커_2004-04-23_일본',
 '돌아온이탈자_1992-05-01_미국',
 '새벽에서밤까지_1990-01-01_한국',
 '하이랜더_1990-04-28_미국',
 '미쉬카_2002-06-20_프랑스',
 '임소로의부마동자_1990-01-01_한국',
 '토요일밤의남자_1992-05-14_미국',
 '사일런트워_2018-05-17_오스트리아 ',
 '감독미카엘하네케_2016-02-25_오스트리아 ',
 '타버린비밀_1990-01-20_오스트리아 ',
 '써스펙트_2004-12-31_미국',
 '구멍_2000-11-30_대만',
 '슈퍼맨일지매_1990-01-01_한국',
 '세븐데이투리브_2001-08-04_독일',
 '돌아온손오공_1990-01-01_한국',
 '핫칙_2003-04-25_미국',
 '쉘부르의우산_1992-09-26_프랑스',
 '아름다운게임_1995-01-01_한국',
 '경복궁1234_1972-05-13_한국',
 '뽀식이와꼬마특공단_1991-01-03_한국',
 '퍼니게임_1997-11-15_오스트리아 ',
 '패배자_1991-01-01_한국',
 '익스큐터:더파이널_2018-01-31_미국',
 '소울어쌔신_2004-06-11_네덜란드',
 '미스터바티뇰_2002-06-20_프랑스',
 '터크에버래스팅_2003-07-04_미국',
 '팝콘_1993-08-21_미국',
 '800_2020-12-10_중국',
 '배꼽버스_1995-12-02_한국',
 '1944_2015-12-23_에스토니아',
 '뮤지엄아워스_2014-01-23_오스트리아 ',
 '이웃집야마다군_2006-06-08_일본',
 '현대고혹자_1996-04-20_홍콩',
 '어페어오브더넥클리스_2002-10-26_미국',
 '옥토퍼스_2000-10-19_미국',
 '언세드_2003-06-20_미국',
 '영구와땡칠이3탄-영구람보_1990-07-28_한국',
 '천방지축_2003-12-12_홍콩',
 '개인교수2012_2012-04-05_벨기에',
 '천국의땅_1990-01-01_한국',
 '아마도내일은_2017-06-01_오스트리아 ',
 '오내사랑임꺽정_1991-01-01_한국',
 '파워킹_1995-07-26_한국',
 '퀵샌드_2005-05-26_프랑스',
 '다케시즈_2006-12-07_일본',
 '장미빛여자_1991-01-01_한국',
 '라스트워리어_1992-09-19_미국',
 '챔프의분노_1992-01-01_한국',
 '데미지_1994-12-17_영국',
 '데쓰위시3_1988-02-18_미국',
 '하나님이어디있어요_1991-01-01_한국',
 '아바타3D스페셜에디션_2010-09-04_미국',
```

'히후데어2:런던테러리스트_2016-08-04_영국',
'포이틱저스티스_1993-07-23_미국',
'스네이크앤이어링_2011-12-01_일본',
'미스터본즈_2004-08-06_남아프리카공화국',
'2424_2002-10-17_한국',
'더벙커:프로젝트12_2017-09-28_스페인',
'우주에서온사나이_1991-07-20_홍콩',
'사랑의여왕_2012-04-12_프랑스',
'미셔너리:감춰진그림자_2016-05-19_미국',
'인피니티_1998-10-16_미국',
'파업전야_1990-03-28_한국',
'여인파티_1989-10-14_한국',
'레더쟈켓_1993-02-06_미국',
'생령_2003-05-30_일본',
'영웅들의날개짓_1993-01-01_한국',
'블리트_2002-06-18_프랑스',
'이유없는반항_2001-01-01_한국',
'세인트_2012-05-10_네덜란드',
'오늘같이좋은날_1991-01-01_한국',
'파라노이드_2006-02-28_영국',
'우주전사불의사나이_1991-01-03_한국',
'센스오브스노우_1998-10-24_스웨덴',
'22_2018-08-14_한국',
'브레이크블레이드:변신의시작_2012-04-19_일본',
'론리플레이스투다이_2012-04-05_영국',
'내가성에관해알고있는몇가지이야기들_1993-01-01_한국',
'프리즈미_2004-08-27_일본',
'하이힐_1992-06-20_스페인',
'굿나잇마미_2016-10-12_오스트리아 ',
'딥코아_2000-10-21_미국',
'경계인들_2018-10-12_한국',
'부킹쏘나타_2001-01-01_한국',
'언피쉬_2000-04-29_오스트리아 ',
'리골레토_2018-06-03_영국',
'주글래살래_2003-03-07_한국',
'루르드_2011-02-17_오스트리아 ',
'하이힐크라임_2002-05-31_미국',
'불타는태양_1990-01-01_한국',
'태권소년어니와마스타김_1989-12-22_한국',
'용감한사람들_1990-01-01_한국',
'리골레토_2016-10-12_프랑스',
'런투유(RUN2U)_2003-05-23_한국',
'달은...해가꾸는꿈_1992-02-29_한국',
'영웅후레쉬_1990-01-01_한국',
'퐁네프의연인들_1992-04-18_프랑스',
'메이_2003-05-02_미국',
'블러드오렌지_2018-10-31_영국',
'캅링크_2001-01-01_한국',
'사보타지2_2001-07-14_미국',
'하드캐쉬_2002-07-19_미국',
'아름다운것들_2002-06-19_프랑스',
'불행한아이의행복_1992-01-01_한국',
'겨울연가_1992-01-01_한국',
'33_2016-04-07_미국',
'만화방_2012-02-29_한국',
'카사블랑카특급_1991-11-09_이탈리아',
'어썰트_2012-05-10_프랑스',
'방세옥2:대도무문_1993-09-25_홍콩',
'데이앤나이트_2013-10-31_오스트리아 ',
'세상밖으로_2007-02-01_일본',
'사우스스트리트의소매치기_2005-07-07_미국',

'불의전차_2016-06-16_영국',
'론울브스_2016-11-24_오스트리아 ',
'버수스_2004-05-28_일본',
'바스키아_1998-09-18_미국',
'별3형제_1977-11-04_한국',
'이터널선샤인_2005-11-10_미국',
'정사3_2005-05-13_핀란드',
'21_2008-06-19_미국',
'디디할리우드_2012-06-21_스페인',
'월트디즈니모험의세계_1979-07-11_미국',
'3일간의전쟁_1993-01-16_미국',
'잉카_2000-10-28_미국',
'몰:데이투킬_2015-09-10_미국',
'하나부사_2004-12-30_일본',
'천녀유혼_1998-01-01_홍콩',
'택시_1998-08-29_스페인',
'총각파티_1992-03-28_미국',
'전갈군단_1991-01-01_한국',
'셜리에관한모든것_2013-12-26_오스트리아 ',
'9/11_2018-01-31_미국',
'해밀턴_2012-04-12_스웨덴',
'화녀_1971-04-01_한국',
'프레디타_2000-06-08_멕시코',
'골뱅이@실명제_2001-01-01_한국',
'타운앤컨트리_2002-11-23_미국',
'품행제로_2004-03-27_프랑스',
'불꽃슛통키_1993-01-01_한국',
'역사는이렇게시작됐다_1992-01-01_한국',
'클럽포보스_2013-12-19_러시아',
'나이트아이2_1993-10-30_미국',
'카운터페이터_2008-07-03_오스트리아 ',
'지금,만나러갑니다_2005-03-25_일본',
'공룡티렉스_2006-04-13_캐나다',
'크리스마스인코네티컷_1992-12-18_미국',
'해피텍사스_2000-08-26_미국',
'해피엑시던트_2003-11-21_미국',
'세나의신혼일기_1992-01-01_한국',
'언두_2005-06-23_일본',
'1987_2017-12-27_한국',
'페이싱_2000-11-18_프랑스',
'벤자민프로젝트_2003-04-11_미국',
'1941_1980-07-26_미국',
'저스틴_2006-12-15_폴란드',
'혈락제_1992-10-24_홍콩',
'슈퍼쇼3D_2011-02-24_한국',
'방콕데인져러스_2001-09-22_태국',
'영웅필살_1990-01-01_한국',
'트레스패스_2012-07-05_미국',
'파리넬리_1995-04-08_프랑스',
'마이퍼스트미스터_2004-12-10_미국',
'이상한나라의앨리스_2018-05-02_영국',
'사랑의이름으로_1992-01-01_한국',
'헤드오버힐스_2003-09-26_미국',
'나인마일즈다운_2012-06-28_영국',
'1408_2007-08-01_미국',
'블랙캣2_1993-09-11_홍콩',
'2012_2009-11-12_미국',
'터커_1992-12-09_미국',
'화엄경_1993-06-26_한국',
'천국의계단_1992-02-15_한국',
'태양유이_2000-07-08_홍콩',

'2010빅뱅라이브콘서트BIGSHOW3D_2011-02-02_한국',
'스네이크온어플레인_2006-12-07_미국',
'토미에-리버스_2005-09-02_일본',
'300_2007-03-14_미국',
'치킨_1996-12-07_오스트리아 ',
'2046_2004-10-15_홍콩',
'계부3_1992-07-11_미국',
'에곤쉴레:욕망이그린그림_2016-12-22_오스트리아 ',
'용궁에서온거북이와이무기_1991-01-01_한국',
'신이버린특공대_2003-08-29_미국',
'어허어이어이가리_1991-01-01_한국',
'게임오버_2001-01-01_한국',
'넝쿨속의히야신스_1992-01-01_한국',
'클래쉬_2011-08-11_베트남',
'잭더베어_1993-09-04_미국',
'옥유단:봉성부인_1992-10-31_홍콩',
'미라클메이커_2004-04-01_영국',
'인포먼트_1998-10-16_아일랜드',
'동거남녀_2003-09-04_홍콩',
'아메리칸파이러브_2004-08-20_미국',
'일트로바토레_2016-08-09_프랑스',
'네이키드트루_1992-11-21_미국',
'세가지사랑,정사_2006-08-03_오스트리아 ',
'1917_2020-02-19_미국',
'정열의카르멘_1992-04-18_스페인',
'섹슈얼이노센스_2000-07-26_미국',
'해저탐험대마린X_1983-02-19_한국',
'하얀전쟁_1992-07-04_한국',
'호프만이야기_2017-07-19_영국',
'프린세스블레이드_2003-10-17_일본',
'휘파람부는여자_1998-02-28_한국',
'가면학원_2001-02-17_일본',
'마술피리_2018-03-04_영국',
'트루저스티스_2011-06-30_미국',
'신경쇠약직전의뱀파이어_2015-05-21_오스트리아 ',
'슬로우페이드_2000-09-30_홍콩',
'토스카_2002-06-19_프랑스',
'추격자_1979-07-13_한국',
'황비홍의서역웅사_1997-02-22_홍콩',
'불안은영혼을잠식한다_1997-11-22_독일',
'경복궁12345_1972-05-13_한국',
'프리처스와이프_1997-02-22_미국',
'이도백화_1995-01-01_한국',
'나이제너를잊으리_1992-01-01_한국',
'라보엠_2018-04-01_영국',
'립이어:자학의성_2017-09-28_프랑스']

In [12]:

```
# 박스오피스 확인: 천녀유혼
df_bo.loc[df_bo["영화명"]=="천녀유혼"]
```

Out[12]:

	순번	영화명	감독	제작사	수입사	배급사	개봉일	영화유형	영화형태	국적	전국스크린수	전국 매출액	전국관객수	서울 매
2741	2742	천녀유혼	엽위신	NaN	NaN	씨너스엔터테인먼트(주)	2011-05-12	개봉영화	장편	중국	240	497089500.0	66219	1616730
21728	21729	천녀유혼	정소동	(주)동아수출공사	NaN	NaN	1987-12-05	개봉영화	장편	홍콩	0	0.0	0	

In [13]:

```
# 영화목록 확인: 천녀유혼
df_movielist.loc[df_movielist["movieNm"]=="천녀유혼"]
```

Out[13]:

	movieCd	movieNm	movieNmEn	prdtYear	openDt	typeNm	prdtStatNm	nationAlt
13048	20080605	천녀유혼	A Chinese Ghost Story	1987	19871205	장편	개봉	홍콩
13946	20110311	천녀유혼	A Chinese Fairy Tale	2010	20110512	장편	개봉	중국
19873	20124708	천녀유혼	A Chinese Ghost Story: The Tsui Hark Animation	1997	19980101	장편	개봉	홍콩,일본

In [14]:

```
# 박스오피스 확인: 품행제로
df_bo.loc[df_bo["영화명"]=="품행제로"]
```

Out[14]:

	순번	영화명	감독	제작사	수입사	배급사	개봉일	영화유형	영화형태	국적	전국스크린수	전국매출액	전국 관객수	서울매출액	서울 관객수	장르	등급	영화구분
524	525	품행제로	조근식	케이엠컬쳐㈜	NaN	영화사청어람(주)	2002-12-26	개봉영화	장편	한국	0	0.0	1694637	0.0	569421	코미디	15세관람가	일반영화

In [15]:

```
# 영화목록 확인: 품행제로
df_movielist.loc[df_movielist["movieNm"]=="품행제로"]
```

Out[15]:

	movieCd	movieNm	movieNmEn	prdtYear	openDt	typeNm	prdtStatNm	nationAlt
12273	20020239	품행제로	Conduct Zero	2002	20021226	장편	개봉	한국
14267	20112810	품행제로	Zero de conduite	1933	20040327	장편	개봉	프랑스

In [16]:

```
df_bo_genres = df_movie_bo.merge(df_bo_genres[["영화명_개봉일_대표국적", "영화명", "개봉일", "서울 관객수", "전국 관객수", "국적", "openYear"]])
df_bo_genres = df_bo_genres.merge(df_genres.drop(["movieNm", "openYear"], axis=1), on="movieCd")
df_bo_genres = df_bo_genres.drop(["movieNm", "openDt", "repNationNm", "영화명_개봉일_대표국적"], axis=1)
print(df_bo_genres.shape)
df_bo_genres.head(3)
```

(23182, 29)

Out[16]:

	movieCd	영화명	개봉일	서울 관객수	전국 관객수	국적	openYear	G_SF	G_가족	G_공연	G_공포	G_기타	G_다큐멘터리	G_드라마
0	20139221	그래비티	2013-10-17	1311034	3227452	미국	2013	1	0	0	0	0	0	1
1	20201002	조제	2020-12-10	45870	186647	한국	2020	0	0	0	0	0	0	1
2	20010264	아멜리에	2001-10-19	157562	0	프랑스	2001	0	0	0	0	0	0	0

In [17]:

```
df_bo.shape
```

Out[17]:

(23271, 19)

In [18]:

```
df_bo_genres.shape
```

Out[18]:

(23182, 29)

In [19]:

```
# 영화 장르, 관객 확인: 천녀유혼
df_bo_genres.loc[df_bo_genres["영화명"]=="천녀유혼"]
```

Out[19]:

	movieCd	영화명	개봉일	서울 관객수	전국 관객수	국적	openYear	G_SF	G_가족	G_공연	G_공포	G_기타	G_다큐멘터리	G_드라마
11569	20080605	천녀유혼	1987-12-05	31639	0	홍콩	1987	0	0	0	1	0	0	1
12433	20110311	천녀유혼	2011-05-12	20447	66219	중국	2011	1	0	0	0	0	0	0

In [20]:

```
# 영화 장르, 관객 확인: 품행제로
df_bo_genres.loc[df_bo_genres["영화명"]=="품행제로"]
```

Out[20]:

	movieCd	영화명	개봉일	서울 관객수	전국 관객수	국적	openYear	G_SF	G_가족	G_공연	G_공포	G_기타	G_다큐멘터리	G
10809	20020239	품행제로	2002-12-26	569421	1694637	한국	2002	0	0	0	0	0	0	

In [21]:

```
# 데이터 백업
df_bo_genres.to_pickle("./data/df_bo_genres.pkl")
```

2.2.2. 년도별 데이터 정리

In [22]:

```
# 년도별 데이터 정리
df_boY = df_bo[["openYear", "전국 관객수", "서울 관객수"]].groupby("openYear").sum().reset_index
()
df_boY.head()

# 한국영화, 해외영화
df_bo_kr = df_bo.query("국적 == '한국'")
df_boY_kr = df_bo_kr[["openYear", "전국 관객수", "서울 관객수"]].groupby("openYear").sum().reset
_index()

df_bo_nkr = df_bo.query("국적 != '한국'")
df_boY_nkr = df_bo_nkr[["openYear", "전국 관객수", "서울 관객수"]].groupby("openYear").sum().res
et_index()
```

In [23]:

```
df_boY_kr.head()
```

Out[23]:

	openYear	전국 관객수	서울 관객수
0	1971	0	3913867
1	1972	0	2845083
2	1973	0	1856398
3	1974	0	2695969
4	1975	0	2129130

In [24]:

```
df_boY_nkr.head()
```

Out[24]:

	openYear	전국 관객수	서울 관객수
0	1971	0	7589223
1	1972	0	7893193
2	1973	0	6471786
3	1974	0	5970750
4	1975	0	4274980

In [25]:

```
# 데이터 백업
df_bo_kr.to_pickle("./data/df_bo_kr.pkl")
df_boY_kr.to_pickle("./data/df_boY_kr.pkl")
df_bo_nkr.to_pickle("./data/df_bo_nkr.pkl")
df_boY_nkr.to_pickle("./data/df_boY_nkr.pkl")
```

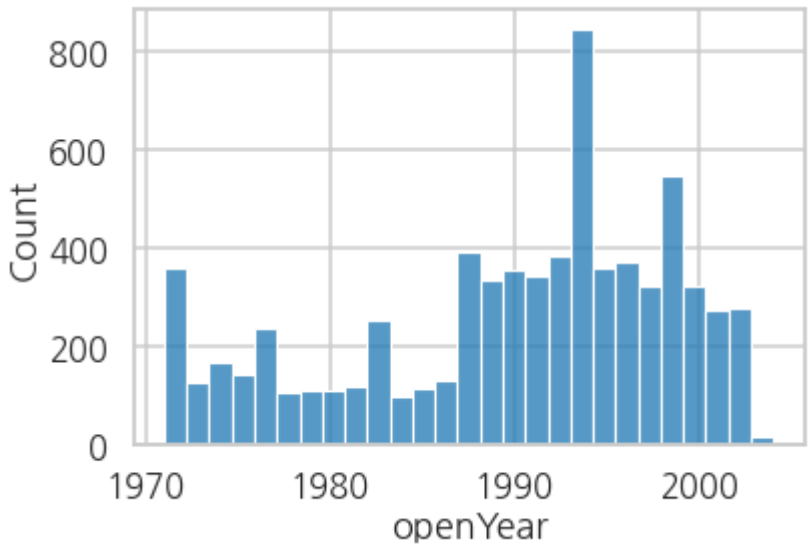
기초분석

In [26]:

```
# 전국 관객 수 없는 영화 확인
sns.histplot(df_bo.loc[df_bo["전국 관객수"]==0, "openYear"])
```

Out[26]:

<AxesSubplot:xlabel='openYear', ylabel='Count'>

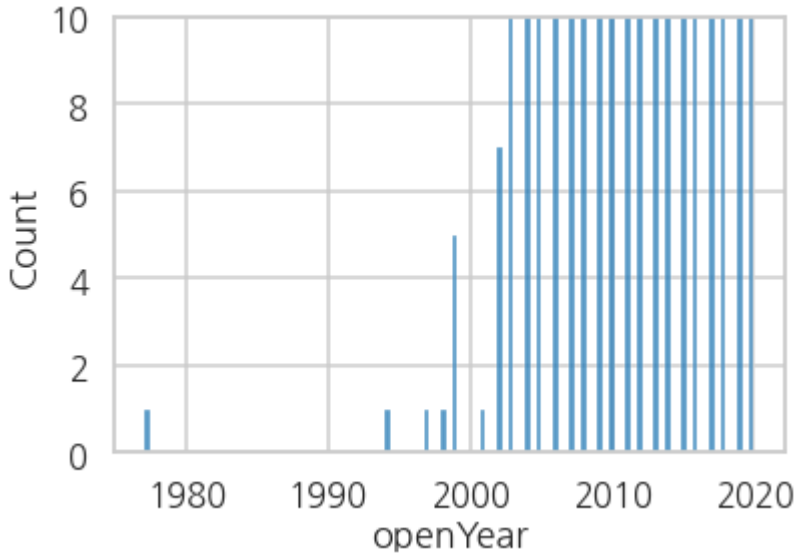


In [27]:

```
# 전국 관객 수 있는 영화 확인
ax = sns.histplot(df_bo.loc[df_bo["전국 관객수"]!=0, "openYear"])
ax.set_ylim(0, 10)
```

Out[27]:

(0.0, 10.0)

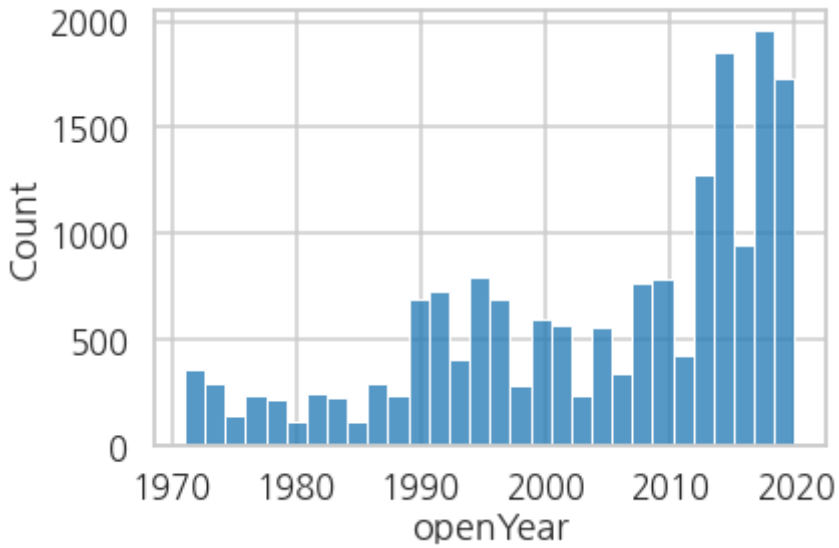


In [28]:

```
# 서울 관객 수 있는 영화 확인
sns.histplot(df_bo.loc[df_bo["서울 관객수"]!=0, "openYear"])
```

Out[28]:

<AxesSubplot:xlabel='openYear', ylabel='Count'>

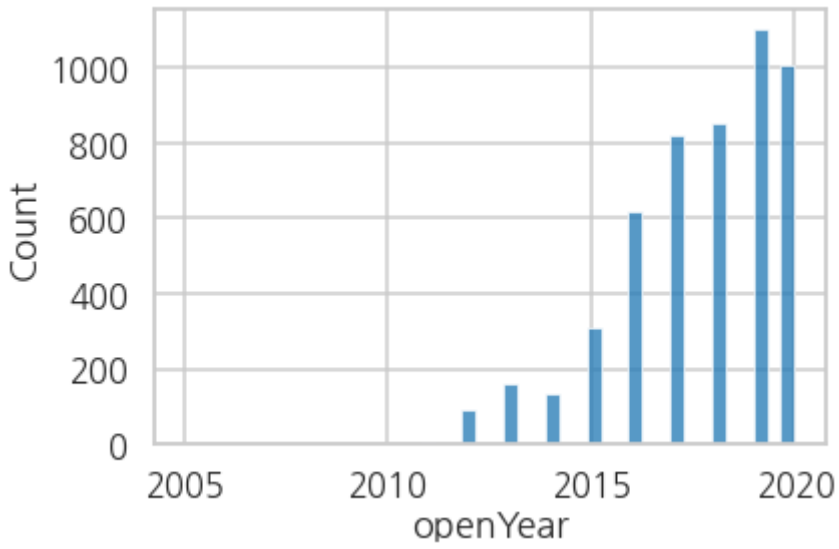


In [29]:

```
# 서울 관객 수 없는 영화 확인
sns.histplot(df_bo.loc[df_bo["서울 관객수"]==0, "openYear"])
```

Out[29]:

<AxesSubplot:xlabel='openYear', ylabel='Count'>



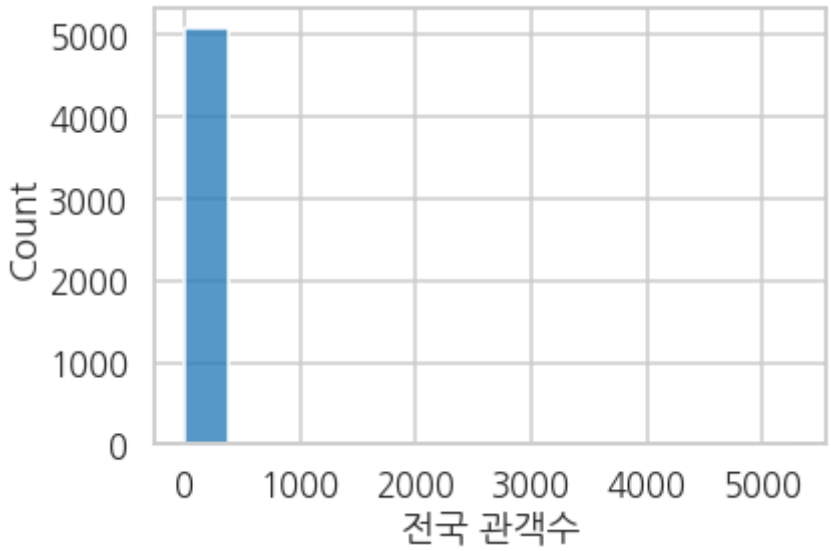
In [30]:

```
# 서울 관객 수 없는 영화의 전국 관객 수
print(df_bo.loc[df_bo["서울 관객수"]==0, "전국 관객수"].max())
sns.histplot(df_bo.loc[df_bo["서울 관객수"]==0, "전국 관객수"])
```

5279

Out[30]:

<AxesSubplot:xlabel='전국 관객수', ylabel='Count'>



In [31]:

```
df_bo.loc[df_bo["서울 관객수"]==0].sort_values("전국 관객수", ascending=False).head(10)
```

Out[31]:

	순번	영화명	감독	제작사	수입사	배급사	개봉일	영화유형	영화형태	국적	전국스크린수	전국 매출액	전국관객수	서울매출액	서울관객수	장르	등급
5116	5117	테레사팔머의감금	데이비드데닌	NaN	(주)컴퍼니엘	(주)컴퍼니엘	2013-11-28	개봉영화	장편	호주	4	31659000.0	5279	0.0	0	스릴러	청소년관람불기
5226	5227	서스펙트제로	E.엘리아스메리지	파라마운트픽쳐스	소니픽쳐스릴리징코리아	소니픽쳐스릴리징코리아	2005-03-17	개봉영화	장편	미국	0	0.0	4933	0.0	0	공포(호러)	15세관람가
5331	5332	제노바	마이클윈터바텀	레볼루션필름	㈜영화사백두대간	㈜영화사백두대간	2009-11-19	개봉영화	장편	영국	1	33567015.0	4450	0.0	0	드라마	15세이상관람가
5583	5584	핀두스이야기	알리사마디아하디	NaN	(주)영화사안다미로	(주)레인메이커필름	2017-11-30	개봉영화	장편	독일	22	17345000.0	3524	0.0	0	애니메이션	전체관람가
5861	5862	릴라와마법의책	마르셀라링콘곤잘레스	NaN	씨네힐	(주)영화사오원	2018-11-08	개봉영화	장편	콜롬비아	24	14141000.0	2698	0.0	0	어드벤처	전체관람가

	순번	영화명	감독	제작사	수입사	배급사	개봉일	영화유형	영화형태	국적	전국스크린수	전국 매출액	전국관객수	서울매출액	서울관객수	장르	등급
6165	6166	파이어위드파이어	데이빗바렛	NaN	(주)소나무픽쳐스	(주)영화사폴	2013-02-21	개봉영화	장편	미국	1	4355000.0	2030	0.0	0	액션	청소년관림불기
6379	6380	사대명포2	진가상	NaN	(주)디앤아이컴퍼니	(주)디앤아이컴퍼니	2014-01-16	개봉영화	장편	홍콩	5	7590000.0	1700	0.0	0	액션	15세이상관림기
6411	6412	셀러브리티섹스테이프	스콧휠러	NaN	(주)스넵쏘울	(주)스넵쏘울	2013-06-27	개봉영화	장편	미국	2	3487000.0	1664	0.0	0	코미디	청소년관림불기
6421	6422	친구의아내를탐하다	얀흐레베이크	NaN	(주)소나무픽쳐스	(주)소나무픽쳐스	2013-05-31	개봉영화	장편	체코	1	3501000.0	1653	0.0	0	코미디	청소년관림불기
6431	6432	아웃포스트 : 블랙선	스티브바커	NaN	(주)소나무픽쳐스	(주)소나무픽쳐스	2013-06-20	개봉영화	장편	영국	1	3495000.0	1650	0.0	0	스릴러	15세이상관림기

In [32]:

```
# 서울 관객 수 없는 영화 목록
df_bo.loc[df_bo["서울 관객수"] == 0]
```

Out[32]:

	순번	영화명	감독	제작사	수입사	배급사	개봉일	영화유형	영화형태	국적	전국스크린수	전국 매출액	전국관객수	서울매출액	서울관객수	장르
5116	5117	테레사팔머의감금	데이비드데닌	NaN	(주)컴퍼니엘	(주)컴퍼니엘	2013-11-28	개봉영화	장편	호주	4	31659000.0	5279	0.0	0	스릴러
5226	5227	서스펙트제로	E.엘리아스메리지	파라마운트픽쳐스	소니픽쳐스릴리징코리아	소니픽쳐스릴리징코리아	2005-03-17	개봉영화	장편	미국	0	0.0	4933	0.0	0	공포(호러)
5331	5332	제노바	마이클윈터바텀	레볼루션필름	㈜영화사백두대간	㈜영화사백두대간	2009-11-19	개봉영화	장편	영국	1	33567015.0	4450	0.0	0	드라마
5583	5584	핀두스이야기	알리사마디아하디	NaN	(주)영화사안다미로	(주)레인메이커필름	2017-11-30	개봉영화	장편	독일	22	17345000.0	3524	0.0	0	애니메이션
5861	5862	릴라와마법의책	마르셀라링콘곤잘레스	NaN	씨네힐	(주)영화사오원	2018-11-08	개봉영화	장편	콜롬비아	24	14141000.0	2698	0.0	0	어드벤처
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...

	순번	영화명	감독	제작사	수입사	배급사	개봉일	영화유형	영화형태	국적	전국스크린수	전국 매출액	전국관객수	서울매출액	서울관객수	장르
16011	16012	S걸의 덫	그레고리 하타나카	NaN	(주)케이알씨지	(주)케이알씨지	2018-04-10	개봉영화	장편	브라질	1	1000.0	1	0.0	0	멜로/로맨스
16012	16013	T-34	알렉세이 시도로프	NaN	㈜코리아스크린	㈜코리아스크린	2019-06-27	개봉영화	장편	러시아	1	5000.0	1	0.0	0	전쟁
16013	16014	VIP 전용 여자	카네다 사토시	NaN	(주)도키엔터테인먼트	(주)도키엔터테인먼트	2016-12-01	개봉영화	장편	일본	1	8000.0	1	0.0	0	성인물(에로)
16014	16015	VJ의 은밀한 사생활	오카다 히로	NaN	(주)영진크리에이티브	(주)영진크리에이티브	2017-12-25	개봉영화	장편	일본	1	6000.0	1	0.0	0	멜로/로맨스
16015	16016	av 감독의 아내	쿠도마 사노리	NaN	(주)도키엔터테인먼트	(주)도키엔터테인먼트	2016-12-22	개봉영화	장편	일본	1	8000.0	1	0.0	0	성인물(에로)

5109 rows × 19 columns

In [33]:

```
# 서울 관객 수가 0인 작품은 지방에서만 개봉한 작품으로 간주. 정상 데이터 처리.
```

2.3. 월별 박스오피스 데이터(2018-2020)

In [34]:

```
# 박스오피스 데이터: KOBIS 데이터를 받아 header 일부 수정
url_boM_2018 = "https://jehyunlee.github.io/about/src/2021_datastory_movie/KOBIS_movie_2018.xlsx"

# 데이터 불러오기
df_boM_2018 = pd.read_excel(url_boM_2018, engine="openpyxl")
```

In [35]:

```
# 박스오피스 데이터: KOBIS 데이터를 받아 header 일부 수정
url_boM_2019 = "https://jehyunlee.github.io/about/src/2021_datastory_movie/KOBIS_movie_2019.xlsx"

# 데이터 불러오기
df_boM_2019 = pd.read_excel(url_boM_2019, engine="openpyxl")
```

In [36]:

```
# 박스오피스 데이터: KOBIS 데이터를 받아 header 일부 수정
url_boM_2020 = "https://jehyunlee.github.io/about/src/2021_datastory_movie/KOBIS_movie_2020.xlsx"

# 데이터 불러오기
df_boM_2020 = pd.read_excel(url_boM_2020, engine="openpyxl")
```

In [37]:

```
df_boM = pd.concat([df_boM_2018, df_boM_2019, df_boM_2020], axis=0)
# 년월 컬럼 생성
df_boM["YearMonth"] = df_boM["년월"].dt.strftime("%Y. %m.")

print(df_boM.shape)
df_boM.head()
```

(36, 16)

Out[37]:

	년월	한국 개봉 편수	한국 상영 편수	한국 매출액	한국 관객수	한국 점유율	외국 개봉 편수	외국 상영 편수	외국 매출액	외국 관객수	외국 점유율
0	2018-01-01	65	129	112741611565	14061620	0.613	94	264	70356350290	8881836	0.387
1	2018-02-01	36	121	56020721704	6989199	0.449	89	269	70393244100	8562561	0.551
2	2018-03-01	62	165	63213457740	7916144	0.618	92	294	40499257340	4885367	0.382
3	2018-04-01	56	154	38011778590	4690560	0.333	90	283	81723369503	9375683	0.667
4	2018-05-01	53	153	42903891874	5094177	0.321	91	293	95910974486	10797538	0.679

기초분석

In [38]:

```
# 한국 점유율, 외국 점유율 계산식? : 점유율/전체 점유율
(df_boM["한국 관객수"]/df_boM["전체 관객수"])[:5]
```

Out[38]:

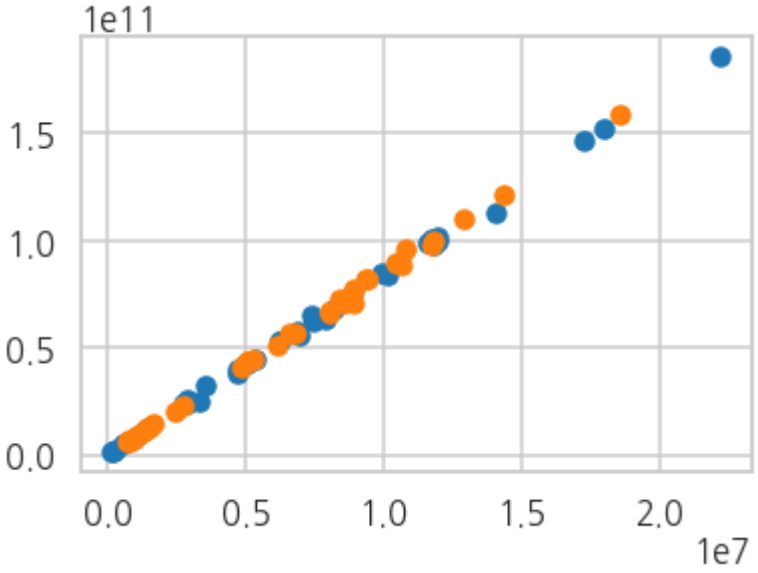
```
0    0.612882
1    0.449415
2    0.618376
3    0.333462
4    0.320556
dtype: float64
```

In [39]:

```
# 매출액 vs 관객수

plt.scatter(df_boM["한국 관객수"], df_boM["한국 매출액"])
plt.scatter(df_boM["외국 관객수"], df_boM["외국 매출액"])
```

Out[39]:

```
<matplotlib.collections.PathCollection at 0x7fb64ef6d2d0>
```



In [40]:

```
# 한국영화 관객 1인당 매출: 8366원
(df_boM["한국 매출액"]/df_boM["한국 관객수"]).mean()
```

Out[40]:

```
8365.519913706084
```

In [41]:

```
# 해외영화 관객 1인당 매출: 8429원
(df_boM["외국 매출액"]/df_boM["외국 관객수"]).mean()
```

Out[41]:

```
8429.49551280481
```

3. 데이터 시각화

In [42]:

```
# 이미지 저장 폴더
!mkdir images
```

3.1. 색상 설정

In [43]:

```
# 국가별 색상코드
c_kr = "darkblue"       # 한국
c_etc = "0.7"           # 기타, 해외
c_it = "g"              # 이탈리아
c_cn = "darkred"        # 중국
c_fr = "gold"           # 프랑스
c_hk = "orangered"      # 홍콩
c_gb = "orchid"         # 영국
c_jp = "thistle"        # 일본
c_us = "mediumpurple"   # 미국

# 국가별 색상
N_colors = dict(zip(["한국", "기타", "해외", "미국", "일본", "홍콩", "프랑스", "영국"],
                    [c_kr, c_etc, c_etc, c_us, c_jp, c_hk, c_fr, c_gb]))
```

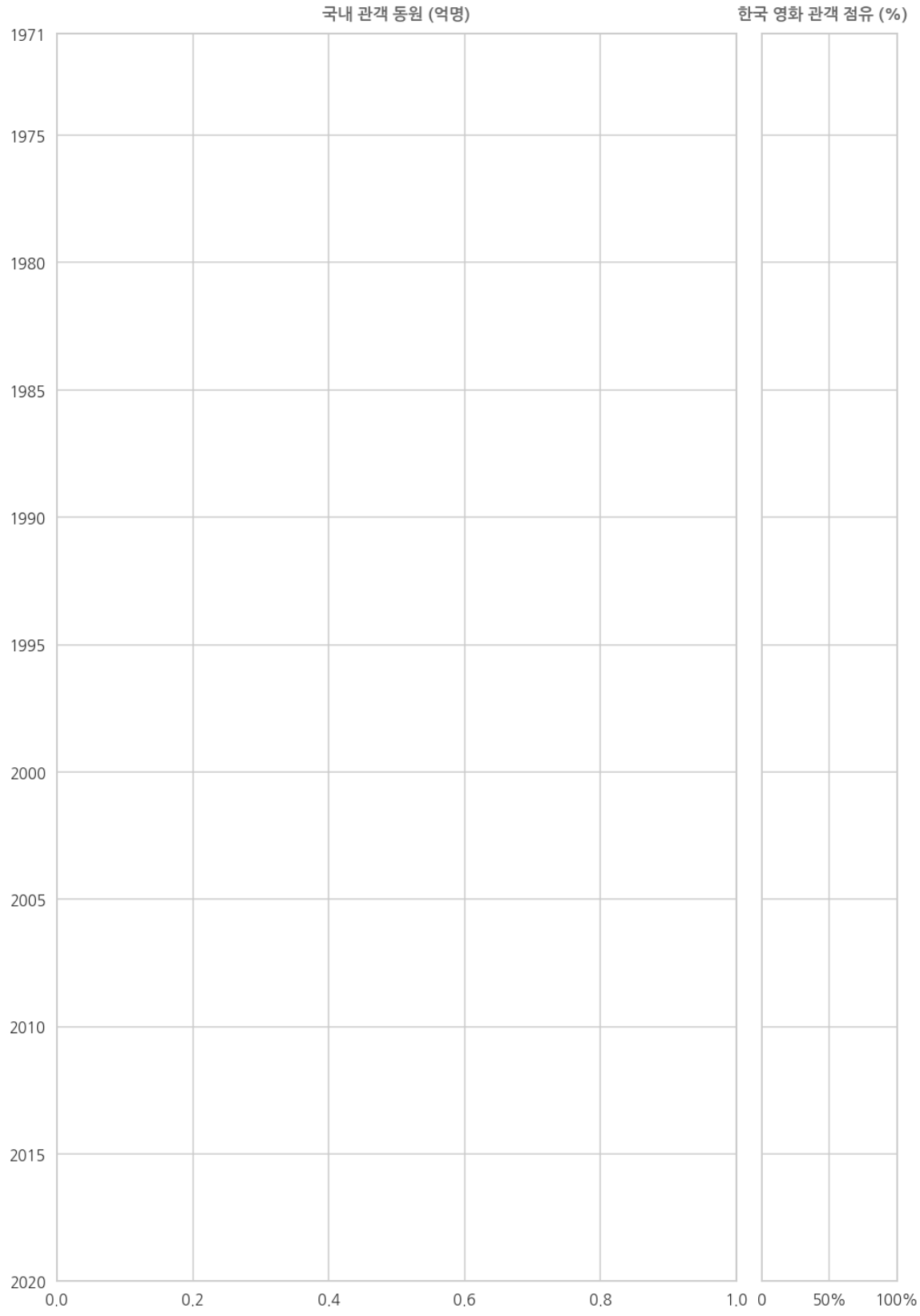
In [44]:

```
# 장르별 색상코드
cmap = plt.get_cmap("tab20")

c_drama = cmap(0/20)
c_romance = cmap(1/20)
c_action = cmap(2/20)
c_comedy = cmap(3/20)
c_thriller = cmap(12/20)
c_ero = cmap(13/20)
c_horror = cmap(6/20)
c_crime = cmap(7/20)
c_ani = cmap(8/20)
c_adv = cmap(9/20)
c_sf = cmap(10/20)
c_fantasy = cmap(11/20)
c_mistery = cmap(4/20)
c_docu = cmap(5/20)
c_family = cmap(14/20)
c_history = cmap(15/20)
c_war = cmap(16/20)
c_play = cmap(17/20)
c_musical = cmap(18/20)
c_western = cmap(19/20)

c_genres = [c_drama, c_romance, c_action, c_comedy, c_thriller, c_ero, c_horror, c_crime, c_ani,
c_adv, c_sf, c_fantasy, c_mistery, c_docu, c_family, c_history, c_war, c_play, c_musical, c_west
ern, c_etc]

# 장르 목록
genres_order = ['드라마', '멜로/로맨스', '액션', '코미디', '스릴러', '성인물', '공포', '범죄',
'애니메이션',
                '어드벤처', 'SF', '판타지', '미스터리', '다큐멘터리', '가족', '사극', '전쟁',
'공연', '뮤지컬', '서부극', '기타']

# 장르별 색상
G_colors = dict(zip([f"G_{g}" for g in genres_order], c_genres))
```

3.2. 서울/전국 관객 동원

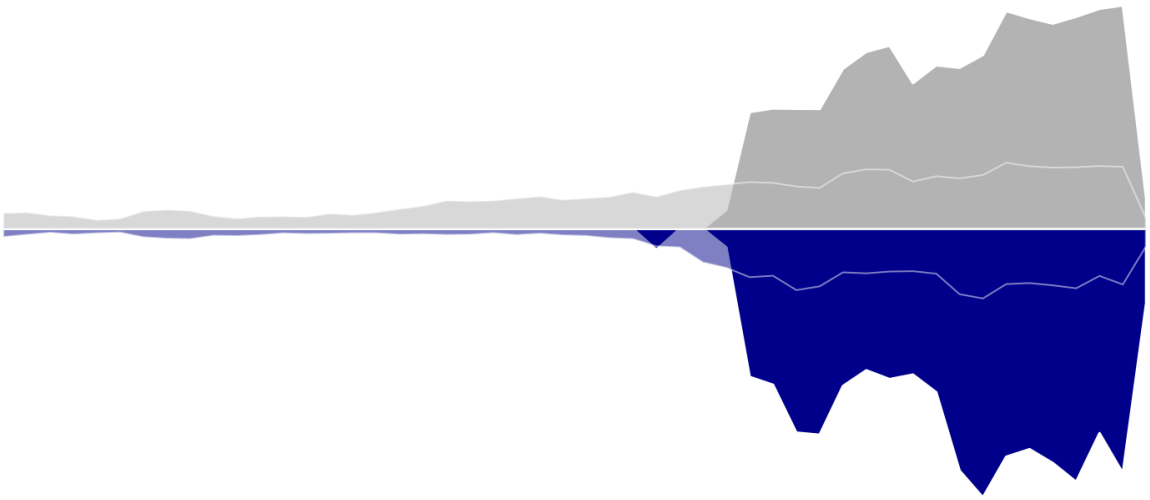
In [45]:

```
fig, axs = plt.subplots(ncols=2, figsize=(14, 20), gridspec_kw={"width_ratios": [5, 1]}, constra
ined_layout=True, sharey=True)

font_title = {"fontweight":"bold", "color":"0.4"}

axs[0].set_title("국내 관객 동원 (억명)", fontdict=font_title, pad=16)
axs[1].set_title("한국 영화 관객 점유 (%)   ", fontdict=font_title, pad=16)
portion_aspect0 = axs[0].get_position().height/axs[0].get_position().width
portion_aspect1 = axs[1].get_position().height/axs[1].get_position().width

yticks = [1971] + list(range(1975, 2025, 5))
axs[0].set_ylim(2020, 1971)
axs[0].set_yticks(yticks)
axs[0].set_yticklabels(yticks)

for ax in axs[1:]:
    ax.set_xticks([0, 0.5, 1])
    ax.set_xticklabels([0, "50%", "100%"])
```


국내 관객 동원 (억명)
한국 영화 관객 점유 (%)
1971
1975
1980
1985
1990
1995
2000
2005
2010
2015
2020
0.0
0.2
0.4
0.6
0.8
1.0
0
50%
100%

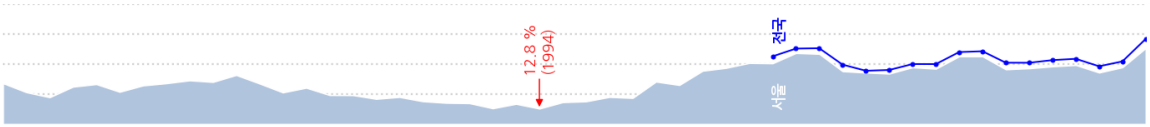
In [46]:

```
fig_p0, ax_p0 = plt.subplots(figsize=(axs[0].get_position().height * 20, axs[0].get_position().width * 14), constrained_layout=True)

ax_p0.fill_between(df_boY_kr["openYear"], -df_boY_kr["전국 관객수"]/1e8, 0, label="한국 영화 전국 관객수", fc=c_kr, lw=3)
ax_p0.fill_between(df_boY_kr["openYear"], -df_boY_kr["서울 관객수"]/1e8, 0, label="한국 영화 서울 관객수", fc=c_kr, lw=2, alpha=0.5)

ax_p0.fill_between(df_boY_nkr["openYear"], df_boY_nkr["전국 관객수"]/1e8, 0, label="해외 영화 전국 관객수", fc=c_etc, lw=3)
ax_p0.fill_between(df_boY_nkr["openYear"], df_boY_nkr["서울 관객수"]/1e8, 0, label="해외 영화 서울 관객수", fc=c_etc, lw=2, alpha=0.5)

ax_p0.grid(False)
ax_p0.set_xlim(1971, 2020)
ax_p0.set_ylim(-1.5, 1.5)
yticks = np.linspace(-1.5, 1.5, 7)
ax_p0.set_yticks(yticks)
ax_p0.set_yticklabels([abs(y) for y in yticks])

ax_p0.axis(False)
fig_p0.savefig("./images/bo_year0.png", dpi=200)
```



In [47]:

```
fig_p1, ax_p1 = plt.subplots(figsize=(axs[1].get_position().height * 20, axs[1].get_position().w
idth * 14), constrained_layout=True)

ax_p1.fill_between(df_boY_kr["openYear"],
                   df_boY_kr["서울 관객수"]/df_boY["서울 관객수"] * 100, 0,
                  fc="lightsteelblue")
ax_p1.plot(df_boY_kr.query("openYear >= 2004")["openYear"],
           df_boY_kr.query("openYear >= 2004")["전국 관객수"]/df_boY.query("openYear >= 2004")[
"전국 관객수"] * 100,
           marker="o", ms=5, c="b", alpha=1, lw=2)

for y in [25, 50, 75, 100]:
    ax_p1.axhline(y, c="lightgray", zorder=-1, ls=":")

p_min = (df_boY_kr["서울 관객수"]/df_boY["서울 관객수"]).min()*100
p_argmin = (df_boY_kr["서울 관객수"]/df_boY["서울 관객수"]).argmin()
ax_p1.annotate(f"{p_min:.1f} %₩n({1971+p_argmin})",
               xy=(1971 + p_argmin, p_min), xytext=(1971 + p_argmin, p_min+30), c="r",
               arrowprops={"arrowstyle":'-|>', "ec":"r", "fc":"r"}, rotation=90, ha="center"
              )

ax_p1.text(2004, 15, "서울", c="w", fontweight="bold", rotation=90, ha="left")
ax_p1.text(2004, 70, "전국", c="b", fontweight="bold", rotation=90, ha="left")

ax_p1.set_xlim(1971, 2020)
ax_p1.set_ylim(0, 100)
ax_p1.axis(False)

fig_p1.savefig("./images/bo_year1.png", dpi=200)
```



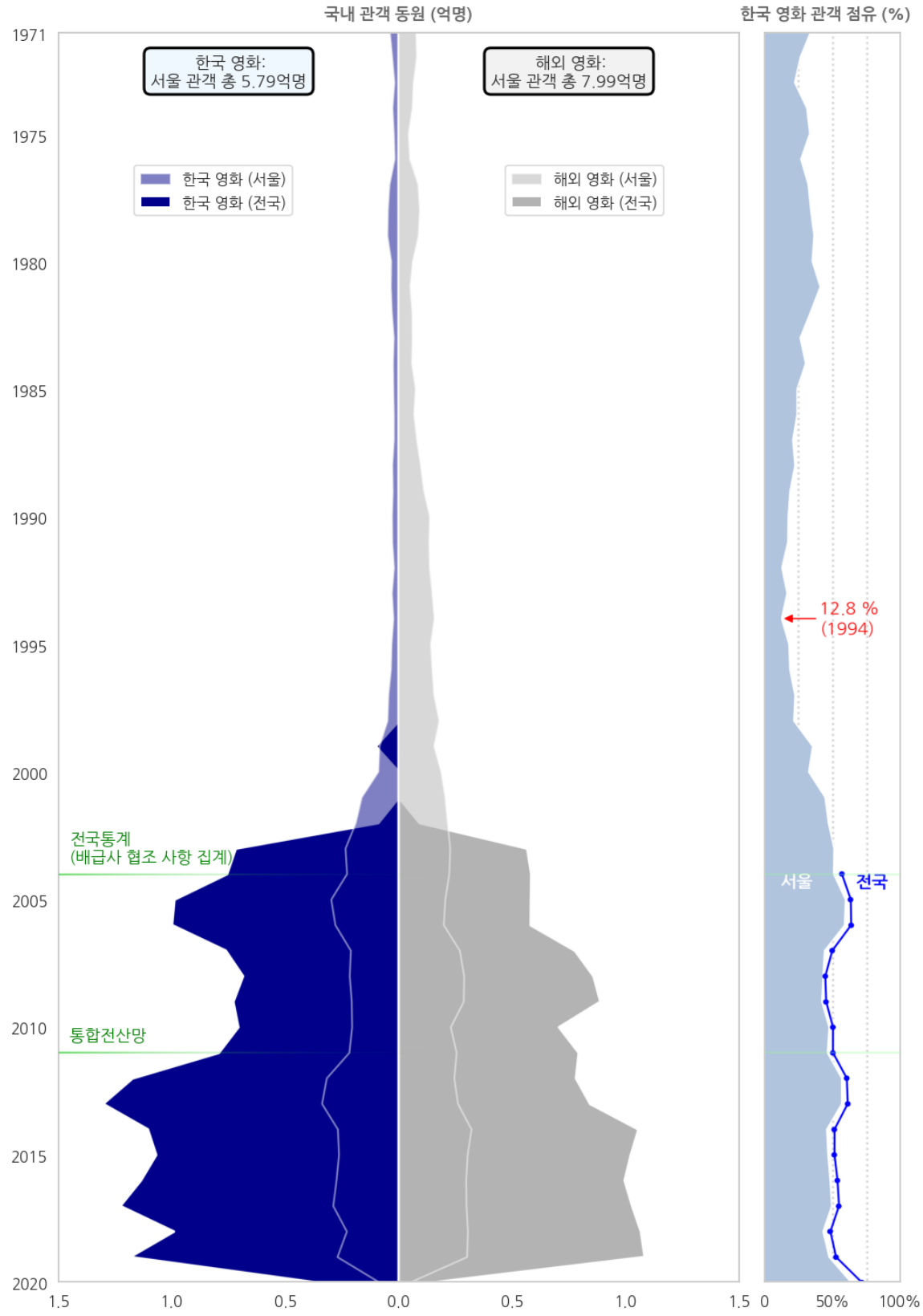
In [48]:

```
# 역사적 사건들
def plot_history(year, text, text_x_shift=50, text_y=None, text_c="green", text_size="medium",
text_fc="w", text_align="center",
                 c_h0 = "limegreen", c_h1 = "palegreen", alpha_h1=0.7, ax=axs):
    if not text_y:
        text_y = year-0.5

    y_line = [year] * 100

    if np.array(axs == None).any():
        ax = plt.gca()
        x0_line = np.linspace(ax.get_xbound()[1], ax.get_xbound()[0], 100)
        x1_line = 0
        ax_0 = ax
    elif isinstance(ax, np.ndarray):
        x0_line = np.linspace(axs[0].get_xbound()[1], axs[0].get_xbound()[0], 100)
        x1_line = np.linspace(axs[1].get_xbound()[1], axs[1].get_xbound()[0], 100)
        ax_0 = ax[0]
    else:
        x0_line = np.linspace(axs[0].get_xbound()[1], axs[0].get_xbound()[0], 100)
        x1_line = 0
        ax_0 = ax

    # axs[0]
    for i in range(99):
        ax_0.plot(x0_line[i:i+2], y_line[i:i+2], c=c_h0,
                  solid_capstyle='butt', alpha=np.power(np.sin(i/100),6)*2, zorder=15)
    ax_0.text(ax_0.get_xbound()[0]+text_x_shift, text_y, text, c=text_c, fontsize=text_size,
              multialignment=text_align, ha="left",
              bbox={"boxstyle":"square", "pad":0.4, "facecolor":text_fc, "edgecolor":"none", "li
newidth":1})

    # axs[1:]
    if isinstance(x1_line, np.ndarray):
        for ax_ in ax[1:]:
            for i in range(99):
                ax_.plot(x1_line[i:i+2], y_line[i:i+2], c=c_h1,
                         solid_capstyle='butt', alpha=alpha_h1, zorder=15)
```

In [49]:

```
### 조립

# axs[0]
fig0_img = plt.imread("./images/bo_year0.png")
fig0_img = fig0_img.swapaxes(0, 1)[:,::-1, :][10:-10,10:-10,:]
x0, x1 = -1.5, 1.5
y1, y0 = 1971, 2020

axs[0].imshow(fig0_img, extent=[x0, x1, y0, y1])
axs[0].set_aspect(abs(x1-x0)/abs(y1-y0)*portion_aspect0 * 20/14)
axs[0].grid(False)
xticks = np.arange(-1.5, 2, 0.5)
axs[0].set_xticks(xticks)
axs[0].set_xticklabels(abs(xticks))

axs[0].text(-0.75, 1972.5, f" 한국 영화:₩n서울 관객 총 {df_boY_kr['서울 관객수'].sum()/1e8:0.2f}
억명", ha="center", va="center",
            bbox={"boxstyle":"round", "pad":0.4,
                  "facecolor":"aliceblue", "edgecolor":"k", "linewidth":3},
            zorder=20)

axs[0].text(0.75, 1972.5, f" 해외 영화:₩n서울 관객 총 {df_boY_nkr['서울 관객수'].sum()/1e8:0.2f}
억명", ha="center", va="center",
            bbox={"boxstyle":"round", "pad":0.4,
                  "facecolor":"0.95", "edgecolor":"k", "linewidth":3},
            zorder=20)

#legend
axs[0].bar(0, 1, fc=c_kr, lw=2, label="서울", alpha=0.5)
axs[0].bar(0, 1, fc=c_kr, lw=2, label="전국")
axs[0].bar(0, 1, fc=c_etc, lw=2, label="서울", alpha=0.5)
axs[0].bar(0, 1, fc=c_etc, lw=2, label="전국")

handles, labels = axs[0].get_legend_handles_labels()
legend_kr = axs[0].legend(handles=handles[:2], labels=["한국 영화 (서울)", "한국 영화 (전국)"],
                          loc="upper left", bbox_to_anchor=(0.1, 0.9))
legend_nkr = axs[0].legend(handles=handles[2:], labels=["해외 영화 (서울)", "해외 영화 (전국)"],
                          loc="upper right", bbox_to_anchor=(0.9, 0.9))

axs[0].add_artist(legend_kr)


# axs[1]
fig1_img = plt.imread("./images/bo_year1.png")
fig1_img = fig1_img.swapaxes(0, 1)[:,::-1, :][10:-10,10:-10,:]
x0, x1 = 0, 1
y1, y0 = 1971, 2020

axs[1].imshow(fig1_img, extent=[x0, x1, y0, y1])
axs[1].set_aspect(abs(x1-x0)/abs(y1-y0)*portion_aspect1 * 20/14)
axs[1].grid(False)

# 전국통계
plot_history(2004, "전국통계₩n(배급사 협조 사항 집계)", text_x_shift=0.05, text_fc='none', alpha
_h1=0.3, text_align="left",
             ax=axs)

# 통합전산망
plot_history(2011, "통합전산망 ", text_x_shift=0.05, text_fc='none', alpha_h1=0.3,
```

```
            ax=axs)

display(fig)
fig.savefig("./images/bo_year.png", dpi=200)
```


국내 관객 동원 (억명)
한국 영화 관객 점유 (%)
한국 영화:
서울 관객 총 5.79억명
해외 영화:
서울 관객 총 7.99억명
한국 영화 (서울)
한국 영화 (전국)
해외 영화 (서울)
해외 영화 (전국)
1971
1975
1980
1985
1990
1995
2000
2005
2010
2015
2020
12.8 %
(1994)
전국통계
(배급사 협조 사항 집계)
통합전산망
서울
전국
1.5
1.0
0.5
0.0
0.5
1.0
1.5
0
50%
100%

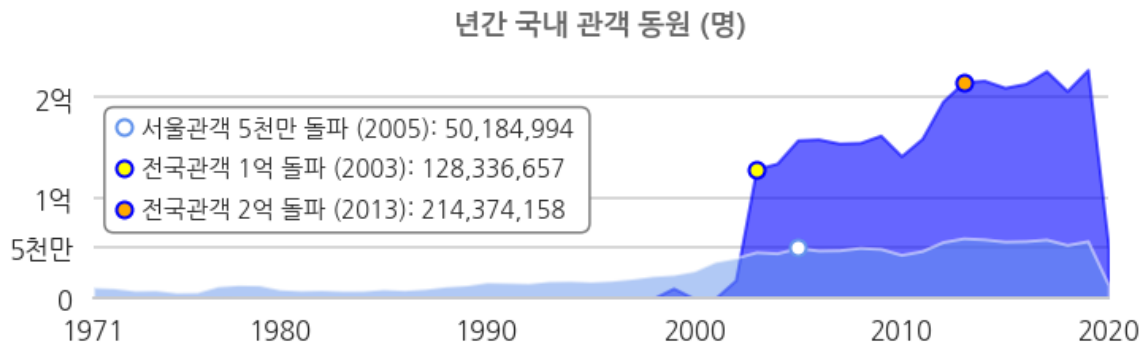
3.3. 서울 관객 5천만, 전국 관객 1, 2억

In [50]:

```
# 서울관객 5천만
seoul_50k_year = df_boY.loc[df_boY["서울 관객수"] > 5e7].iloc[0]["openYear"]
seoul_50k_num = df_boY.loc[df_boY["서울 관객수"] > 5e7].iloc[0]["서울 관객수"]

# 전국관객 1억
nation_100m_year = df_boY.loc[df_boY["전국 관객수"] > 1e8].iloc[0]["openYear"]
nation_100m_num = df_boY.loc[df_boY["전국 관객수"] > 1e8].iloc[0]["전국 관객수"]

# 전국관객 2억
nation_200m_year = df_boY.loc[df_boY["전국 관객수"] > 2e8].iloc[0]["openYear"]
nation_200m_num = df_boY.loc[df_boY["전국 관객수"] > 2e8].iloc[0]["전국 관객수"]
```

In [51]:

```
fig, ax = plt.subplots(figsize=(10, 3), constrained_layout=True)
ax.stackplot(df_boY["openYear"], df_boY["전국 관객수"], colors=["b"], ec="b", lw=2, alpha=0.6)
ax.stackplot(df_boY["openYear"], df_boY["서울 관객수"], colors=["cornflowerblue"], lw=2, alpha=
0.5)
ax.set_xlim(1971, 2020)
ax.set_title("년간 국내 관객 동원 (명)", fontdict=font_title, pad=16)

xticks = [1971, 1980, 1990, 2000, 2010, 2020]
ax.set_xticks(xticks)
ax.set_xticklabels(xticks)
yticks = [0, 50e6, 100e6, 200e6]
ax.set_yticks(yticks)
ax.set_yticklabels([0, "5천만", "1억", "2억"])

# Legend
ax.text(1972, 1.8e8, f"    서울관객 5천만 돌파 ({seoul_50k_year}): {format(seoul_50k_num,
 ',')} ₩n₩n₩n",
        fontsize="small",
        bbox={"boxstyle":"round", "pad":0.4, "facecolor":"w", "edgecolor":"gray"}, ha="left", va
="top")

ax.text(1972, 1.29e8, f"    전국관객 1억 돌파 ({nation_100m_year}): {format(nation_100m_num,
 ',')}",
        fontsize="small", ha="left", va="center")
ax.text(1972, 0.88e8, f"    전국관객 2억 돌파 ({nation_200m_year}): {format(nation_200m_num,
 ',')}",
        fontsize="small", ha="left", va="center")

# 서울관객 5천만
ax.scatter(seoul_50k_year, seoul_50k_num,
           s=100, ec="cornflowerblue", lw=2, c="w", zorder=10)
ax.scatter(1972.5, 1.7e8,
           s=100, ec="cornflowerblue", lw=2, c="w", zorder=10)

# 전국관객 1억
ax.scatter(nation_100m_year, nation_100m_num,
           s=100, ec="blue", lw=2, c="#FFFF00", zorder=10)
ax.scatter(1972.5, 1.29e8,
           s=100, ec="blue", lw=2, c="#FFFF00", zorder=10)
ax.scatter(nation_200m_year, nation_200m_num,
           s=100, ec="blue", lw=2, c="#FFAA00", zorder=10)
ax.scatter(1972.5, 0.88e8,
           s=100, ec="blue", lw=2, c="#FFAA00", zorder=10)


ax.spines[["left", "top", "right"]].set_visible(False)
ax.grid(axis="x")

fig.savefig("./images/bo_year_record.png", dpi=200)
```



3.3. 영화 편당 관객 수

In [52]:

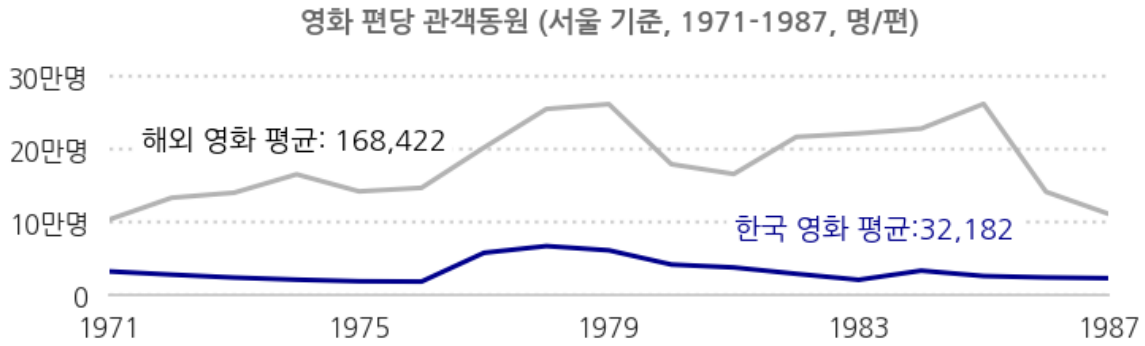
```
df_boY_kr.query('1971 <= openYear <= 1987')['서울 관객수']
```

Out[52]:

```
0      3913867
1      2845083
2      1856398
3      2695969
4      2129130
5      1781146
6      4028277
7      4732529
8      4940554
9      3298231
10     3445652
11     2987014
12     2137983
13     2475661
14     2335820
15     2112474
16     2102287
Name: 서울 관객수, dtype: int64
```

In [53]:

```
## 1971-1987
# 한국영화 1편당 관객 수
num_per_movie_kr = df_boY_kr.query("1971 <= openYear <= 1987")["서울 관객수"]/df_nationsY.query("1971 <= openYear <= 1987")["N_한국"]
mean_kr = format(int(df_boY_kr.query('1971 <= openYear <= 1987')['서울 관객수'].sum()/df_nationsY.query('1971 <= openYear <= 1987')['N_한국'].sum()), ',')

# 해외영화 1편당 관객 수
num_per_movie_nkr = df_boY_nkr.query("1971 <= openYear <= 1987")["서울 관객수"]/df_nationsY.query("1971 <= openYear <= 1987")["해외"]
mean_nkr = format(int(df_boY_nkr.query('1971 <= openYear <= 1987')['서울 관객수'].sum()/df_nationsY.query('1971 <= openYear <= 1987')['해외'].sum()), ',')

fig, ax = plt.subplots(figsize=(10, 3), constrained_layout=True)
ax.plot(range(1971, 1988), num_per_movie_nkr, c=c_etc, lw=3)
ax.plot(range(1971, 1988), num_per_movie_kr, c=c_kr, lw=3)

xticks = [1971, 1975, 1979, 1983, 1987]
ax.set_xticks(xticks)
ax.set_xlim(1971, 1987)
ax.set_ylim(0, 3.5e5)
yticks = [0, 1e5, 2e5, 3e5]
ax.set_yticks(yticks)
ax.set_yticklabels([f"{y/1e4:.0f}만명" if y > 0 else "0" for y in yticks])
for y in yticks:
    ax.axhline(y, c="lightgray", zorder=-1, ls=":")
ax.grid(False)
ax.spines[["left", "top", "right"]].set_visible(False)
# ax.legend(ncol=2, loc="lower right", bbox_to_anchor=(1, 0.8))

ax.text(1971.5, 2e5, f"해외 영화 평균: {mean_nkr}", c="k",
        bbox={"fc":"w"})
ax.text(1981, 8e4, f"한국 영화 평균:{mean_kr}", c=c_kr,
        bbox={"fc":"w"})
ax.set_title("영화 편당 관객동원 (서울 기준, 1971-1987, 명/편)", fontdict=font_title)
fig.savefig("./images/num_movie_mean_1971.png", dpi=200)
```



In [54]:

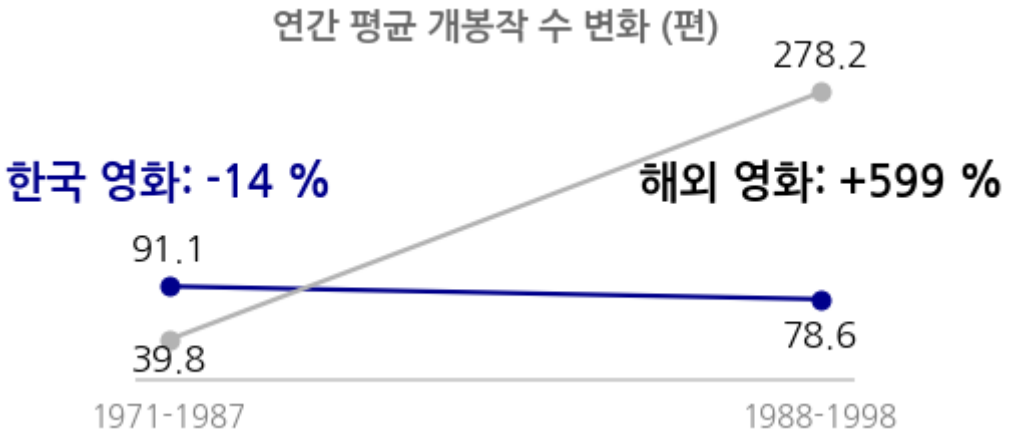
```
168422/32182
```

Out[54]:

5.2334224100428814

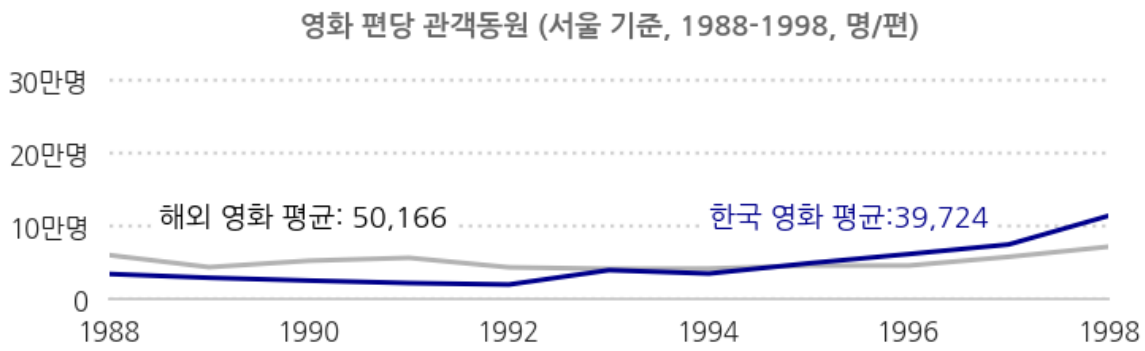
In [55]:

```
## 1988-1998

num_kr_1971 = df_nationsY.query("1971 <= openYear <= 1987")["N_한국"]
num_nkr_1971 = df_nationsY.query("1971 <= openYear <= 1987")["해외"]

num_kr_1988 = df_nationsY.query("1988 <= openYear <= 1998")["N_한국"]
num_nkr_1988 = df_nationsY.query("1988 <= openYear <= 1998")["해외"]

fig, ax = plt.subplots(figsize=(7, 3), constrained_layout=True)
ax.plot([0, 1], [num_kr_1971.mean(), num_kr_1988.mean()], marker="o", color=c_kr)
ax.plot([0, 1], [num_nkr_1971.mean(), num_nkr_1988.mean()], marker="o", color=c_etc)
xticks = [0, 1]
ax.set_xticks(xticks)
ax.set_xticklabels([])
ax.set_title("연간 평균 개봉작 수 변화 (편)", fontdict=font_title, pad=16)
ax.spines[["left", "top", "right"]].set_visible(False)
ax.grid(False)
ax.set_yticks([])

ax.text(0,-20, "1971-1987", ha="center", va="top",
        fontsize="small", c="gray", transform=ax.transData)
ax.text(1,-20, "1988-1998", ha="center", va="top",
        fontsize="small", c="gray", transform=ax.transData)


ax.text(0, num_kr_1971.mean()+20, f"{num_kr_1971.mean():.1f}",
        ha="center", va="bottom", c="k")
ax.text(1, num_kr_1988.mean()-20, f"{num_kr_1988.mean():.1f}",
        ha="center", va="top", c="k")

ax.text(0, num_nkr_1971.mean()-5, f"{num_nkr_1971.mean():.1f}",
        ha="center", va="top", c="k")
ax.text(1, num_nkr_1988.mean()+20, f"{num_nkr_1988.mean():.1f}",
        ha="center", va="bottom", c="k")

ax.text(0, 180,
        f"한국 영화: {num_kr_1988.mean()/num_kr_1971.mean()*100 - 100:.0f} %",
        fontsize="large", fontweight="bold", ha="center", color=c_kr)

ax.text(1, 180,
        f"해외 영화: +{num_nkr_1988.mean()/num_nkr_1971.mean()*100 - 100:.0f} %",
        fontsize="large", fontweight="bold", ha="center", color="k")

ax.set_ylim(0, 300)
fig.savefig("./images/num_change_1988.png", dpi=200)
```


연간 평균 개봉작 수 변화 (편)
278.2
한국 영화: -14 %
해외 영화: +599 %
91.1
78.6
39.8
1971-1987
1988-1998

In [56]:

```
## 1988-1998

year0 = 1988
year1 = 1998

# 한국영화 1편당 관객 수
num_per_movie_kr = df_boY_kr.query(f"{year0} <= openYear <= {year1}")["서울 관객수"]/df_nation
sY.query(f"{year0} <= openYear <= {year1}")["N_한국"]
mean_kr = format(int(df_boY_kr.query(f'{year0} <= openYear <= {year1}')['서울 관객수'].sum()/d
f_nationsY.query(f'{year0} <= openYear <= {year1}')['N_한국'].sum()), ',')

# 해외영화 1편당 관객 수
num_per_movie_nkr = df_boY_nkr.query(f"{year0} <= openYear <= {year1}")["서울 관객수"]/df_nati
onsY.query(f"{year0} <= openYear <= {year1}")["해외"]
mean_nkr = format(int(df_boY_nkr.query(f'{year0} <= openYear <= {year1}')['서울 관객수'].sum()
/df_nationsY.query(f'{year0} <= openYear <= {year1}')['해외'].sum()), ',')

fig, ax = plt.subplots(figsize=(10, 3), constrained_layout=True)
ax.plot(range(year0, year1+1), num_per_movie_nkr, c=c_etc, lw=3)
ax.plot(range(year0, year1+1), num_per_movie_kr, c=c_kr, lw=3)

# xticks = [1971, 1975, 1980, 1985, 1987]
# ax.set_xticks(xticks)
ax.set_xlim(year0, year1)
ax.set_ylim(0, 3.5e5)
yticks = [0, 1e5, 2e5, 3e5]
ax.set_yticks(yticks)
ax.set_yticklabels([f"{y/1e4:.0f}만명" if y > 0 else "0" for y in yticks])
for y in yticks:
    ax.axhline(y, c="lightgray", zorder=-1, ls=":")
ax.grid(False)
ax.spines[["left", "top", "right"]].set_visible(False)
# ax.legend(ncol=2, loc="lower right", bbox_to_anchor=(1, 0.8))

ax.text(year0 +0.5, 1e5, f"해외 영화 평균: {mean_nkr}", c="k",
        bbox={"fc":"w"})
ax.text(year1-4, 1e5, f"한국 영화 평균:{mean_kr}", c=c_kr,
        bbox={"fc":"w"})
ax.set_title(f"영화 편당 관객동원 (서울 기준, {year0}-{year1}, 명/편)", fontdict=font_title)
fig.savefig(f"./images/num_movie_mean_{year0}.png", dpi=200)
```



In [57]:

```
50166/39724
```

Out[57]:

1.262863759943611

In [58]:

```
# 한국영화 1편당 관객 수 (1988-1992)
year0, year1 = 1988, 1992
num_per_movie_kr = df_boY_kr.query(f"{year0} <= openYear <= {year1}")["서울 관객수"]/df_nation
sY.query(f"{year0} <= openYear <= {year1}")["N_한국"]
mean_kr = format(int(df_boY_kr.query(f'{year0} <= openYear <= {year1}')['서울 관객수'].sum()/d
f_nationsY.query(f'{year0} <= openYear <= {year1}')['N_한국'].sum()), ',')
print(mean_kr)
```

25,981

In [59]:

```
# 한국영화 1편당 관객 수 (1993-1995)
year0, year1 = 1993, 1995
num_per_movie_kr = df_boY_kr.query(f"{year0} <= openYear <= {year1}")["서울 관객수"]/df_nation
sY.query(f"{year0} <= openYear <= {year1}")["N_한국"]
mean_kr = format(int(df_boY_kr.query(f'{year0} <= openYear <= {year1}')['서울 관객수'].sum()/d
f_nationsY.query(f'{year0} <= openYear <= {year1}')['N_한국'].sum()), ',')
print(mean_kr)
```

41,515

In [60]:

```
# 한국영화 1편당 관객 수 (1988-1995)
year0, year1 = 1988, 1995
num_per_movie_kr = df_boY_kr.query(f"{year0} <= openYear <= {year1}")["서울 관객수"]/df_nation
sY.query(f"{year0} <= openYear <= {year1}")["N_한국"]
mean_kr = format(int(df_boY_kr.query(f'{year0} <= openYear <= {year1}')['서울 관객수'].sum()/d
f_nationsY.query(f'{year0} <= openYear <= {year1}')['N_한국'].sum()), ',')
print(mean_kr)
```

30,419

In [61]:

```
# 한국영화 1편당 관객 수 (1996-1998)
year0, year1 = 1996, 1998
num_per_movie_kr = df_boY_kr.query(f"{year0} <= openYear <= {year1}")["서울 관객수"]/df_nation
sY.query(f"{year0} <= openYear <= {year1}")["N_한국"]
mean_kr = format(int(df_boY_kr.query(f'{year0} <= openYear <= {year1}')['서울 관객수'].sum()/d
f_nationsY.query(f'{year0} <= openYear <= {year1}')['N_한국'].sum()), ',')
print(mean_kr)
```

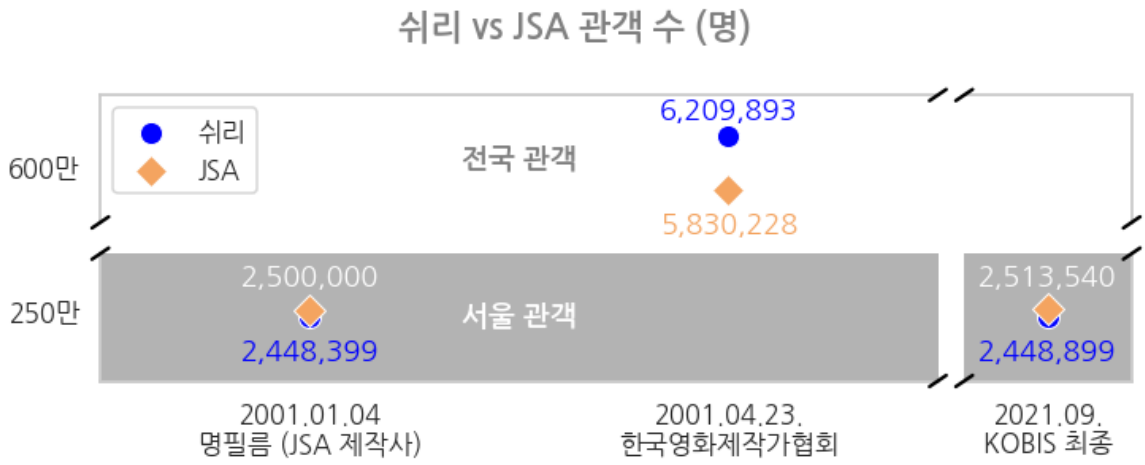
81,360

3.5. 쉬리 vs JSA

In [62]:

```
sj_x = ["2001.01.04₩n명필름 (JSA 제작사)", "2001.04.23.₩n한국영화제작가협회", "2021.09.₩nKOBIS
 최종"]
sj_s = [2.448399, 6.209893, 2.448899]
sj_j = [2.500000, 5.830228, 2.513540]

fig = plt.figure(figsize=(10, 4), constrained_layout=True)
axd = fig.subplot_mosaic(
    """
    AC
    BD
    """,
gridspec_kw={"width_ratios":[2, 0.4], "wspace":0.01, "hspace":0.01})

#
c_s = "blue"
c_j = "sandybrown"
axd["A"].scatter(sj_x[:2], sj_s[:2], marker="o", s=200, c=c_s, ec="w", lw=1, label="쉬리")
axd["A"].scatter(sj_x[:2], sj_j[:2], marker="D", s=200, c=c_j, ec="w", lw=1, label="JSA")
axd["D"].scatter(sj_x[2], sj_s[2], marker="o", s=200, c=c_s, ec="w", lw=1)
axd["D"].scatter(sj_x[2], sj_j[2], marker="D", s=200, c=c_j, ec="w", lw=1)
axd["A"].set_xlim(-0.5, 1.5)
axd["A"].set_ylim(5.600000, 6.500000)
axd["C"].set_ylim(5.600000, 6.500000)
axd["A"].set_yticks([6])
axd["A"].set_yticklabels(["600만"])
axd["A"].set_xticks([])
axd["C"].set_yticks([])
axd["A"].grid(False)
axd["C"].grid(False)
axd["C"].set_xticks([])

axd["B"].scatter(sj_x[:2], sj_s[:2], marker="o", s=200, c=c_s, ec="w", lw=1)
axd["B"].scatter(sj_x[:2], sj_j[:2], marker="D", s=200, c=c_j, ec="w", lw=1)
axd["B"].set_yticks([2.5])
axd["B"].set_yticklabels(["250만"])
axd["B"].set_xlim(-0.5, 1.5)
axd["B"].set_ylim(2, 2.90000)
axd["D"].set_ylim(2, 2.90000)
axd["D"].set_yticks([])
axd["C"].set_xlim(-0.5, 0.5)
axd["D"].set_xlim(-0.5, 0.5)
axd["B"].grid(False)
axd["D"].grid(False)

axd["A"].spines[["bottom", "right"]].set_visible(False)
axd["B"].spines[["top", "right"]].set_visible(False)
axd["C"].spines[["bottom", "left"]].set_visible(False)
axd["D"].spines[["top", "left"]].set_visible(False)

axd["B"].set_facecolor("0.7")
axd["D"].set_facecolor("0.7")


fig.suptitle("쉬리 vs JSA 관객 수 (명)₩n", fontweight="bold", color="gray")
axd["B"].text(sj_x[0], sj_s[0]-0.15, format(int(sj_s[0]*1e6), ","), va="top", ha="center", color
=c_s)
axd["B"].text(sj_x[0], sj_s[0]+0.2, format(int(sj_j[0]*1e6), ","), va="bottom", ha="center", col
or="w")
axd["D"].text(sj_x[2], sj_s[2]-0.15, format(int(sj_s[2]*1e6), ","), va="top", ha="center", color
```

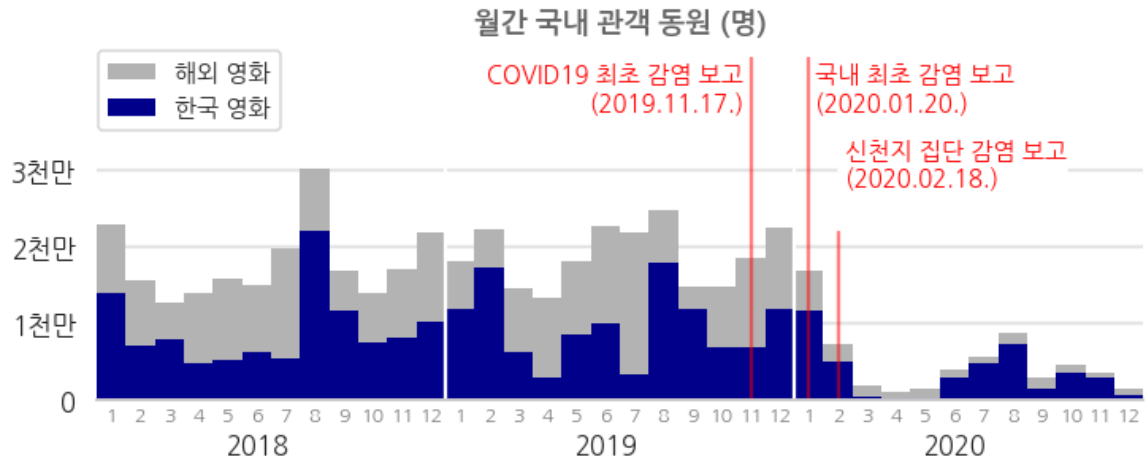
```
=c_s)
axd["D"].text(sj_x[2], sj_s[2]+0.2, format(int(sj_j[2]*1e6), ","), va="bottom", ha="center", col
or="w")
axd["A"].text(sj_x[1], sj_s[1]+0.1, format(int(sj_s[1]*1e6), ","), va="bottom", ha="center", col
or=c_s)
axd["A"].text(sj_x[1], sj_j[1]-0.15, format(int(sj_j[1]*1e6), ","), va="top", ha="center", color
=c_j)

def plot_slash(ax_key, x, y, dx=0.02, dy=0.04):
    kwargs = dict(color='k', clip_on=False)
    axd[ax_key].plot((x-dx, x+dx), (y-dy, y+dy), **kwargs, zorder=3)
    return axd[ax_key]
plot_slash("A", axd["A"].get_xbound()[0], axd["A"].get_ybound()[0])
plot_slash("A", axd["A"].get_xbound()[1], axd["A"].get_ybound()[1])
plot_slash("B", axd["B"].get_xbound()[0], axd["B"].get_ybound()[1])
plot_slash("B", axd["B"].get_xbound()[1], axd["B"].get_ybound()[0])
plot_slash("C", axd["C"].get_xbound()[0], axd["C"].get_ybound()[1], dx=0.05)
plot_slash("C", axd["C"].get_xbound()[1], axd["C"].get_ybound()[0], dx=0.05)
plot_slash("D", axd["D"].get_xbound()[0], axd["D"].get_ybound()[0], dx=0.05)
plot_slash("D", axd["D"].get_xbound()[1], axd["D"].get_ybound()[1], dx=0.05)

axd["A"].text(0.5, 6, "전국 관객", fontweight="bold", color="gray", ha="center")
axd["B"].text(0.5, 2.4, "서울 관객", fontweight="bold", color="w", ha="center")

axd["A"].legend(loc="upper left")

fig.savefig("./images/swiri_jsa.png", dpi=200)
```



3.4. 코로나19 효과

In [233]:

```
fig, ax = plt.subplots(figsize=(10, 4), constrained_layout=True)
ax.bar(df_boM["YearMonth"], df_boM["전체 관객수"], width=1, ec='none',
       color=[c_etc]*36, label="해외 영화")
ax.bar(df_boM["YearMonth"], df_boM["한국 관객수"], width=1, ec='none',
       color=[c_kr]*36, label="한국 영화")

for y in range(3):
    for m in range(12):
        ax.text(12*y + m, -1e6, f"{m+1}", fontsize="x-small", c="gray", ha="center", va="top")

ax.set_xlim(-0.5, 35.5)
ax.set_xticks([5, 17, 29])
ax.set_xticklabels([2018, 2019, 2020], va="top")
ax.tick_params(axis="x", pad=12)
ax.set_ylim(0, 4.5e7)
yticks=np.arange(0, 4e7, 1e7)
ax.set_yticks(yticks)
ax.set_yticklabels([f"{y/1e7:.0f}천만" if y > 0 else "0" for y in yticks])
ax.grid(False)

ax.set_title("월간 국내 관객 동원 (명)", fontdict=font_title, pad=16)
ax.spines[["left", "top", "right"]].set_visible(False)
for y in yticks:
    ax.axhline(y, c="gray", zorder=-1, alpha=0.2)
ax.axvline(11.5, c="w")
ax.axvline(23.5, c="w")

# 발병
ax.axvline(22, c="r", alpha=0.5)
ax.axvline(24, c="r", alpha=0.5)
ax.plot([25, 25], [0, 2.2e7], c="r", alpha=0.5)
ax.text(22, 3.8e7, "COVID19 최초 감염 보고 Wn(2019.11.17.) ", ha="right", fontsize="small", c=
"r",
        bbox={"pad":0, "fc":"w"}, zorder=-0.5)
ax.text(24, 3.8e7, " 국내 최초 감염 보고Wn (2020.01.20.)", ha="left", fontsize="small", c="r",
        bbox={"pad":0, "fc":"w"}, zorder=-0.5)
ax.text(25, 2.8e7, " 신천지 집단 감염 보고Wn (2020.02.18.)", ha="left", fontsize="small", c="r",
        bbox={"pad":0, "fc":"w"}, zorder=-0.5)

fig.legend(loc="lower left", bbox_to_anchor=(0.07, 0.7), fontsize="small")

fig.savefig("./images/bo_covid19.png", dpi=200)
```



3.4. 년도별 서울 관객 분포

3.4.1. 시각화 코드

In [64]:

```
def plot_boS_year(year0, year1, nation, xmax=None, annot_pos=0.8, legend=False):

    # 영화 편당 서울 관객 수 분포
    if nation == '해외':
        df_bo_distS = df_bo_genres.query("국적 != '한국'").groupby("openYear").agg(list)["서울
 관객수"].reset_index()
        if not xmax:
            xmax = df_bo.query(f"{year0} <= openYear <= {year1}").query(f"국적 != '한국'")["서
울 관객수"].max()

    else:
        df_bo_distS = df_bo_genres.query(f"국적 == '{nation}'").groupby("openYear").agg(list)[
"서울 관객수"].reset_index()
        if not xmax:
            xmax = df_bo.query(f"{year0} <= openYear <= {year1}").query(f"국적 == '{nation}'"
)["서울 관객수"].max()


    # 국가별 색상
    c_nation = N_colors[nation]

    nrows = year1-year0+1
    fig, axes = plt.subplots(nrows=nrows, ncols=2, figsize=(12, 0.7*nrows),
                             # sharex=True,
                             # sharey=True,
                             # gridspec_kw={"hspace":-0.5},
                             # gridspec_kw={"width_ratios":[3,1]},
                             # constrained_layout=True
                              )

    # 왼쪽. KDE plot
    z=0
    for ax, year in zip(axes[:,0], range(year0, year1+1)):
        ax.grid(False)
        ax.set_yticks([0])
        ax.set_yticklabels([""])
        ax.text(0, 0, f"{year} ", transform=ax.transAxes, ha="right", va="bottom")
        ax.set_ylabel("")
        ax.set_zorder(z+10)
        ax.set_facecolor("none")
        ax.spines[["left", "top", "right"]].set_visible(False)

        list_bo_year_ = df_bo_distS.query(f"openYear == {year}")["서울 관객수"].values
        if len(list_bo_year_) == 0:
            ax.set_ylim(0, np.finfo(float).eps)
            continue

        list_bo_year = list_bo_year_[0]
        if len(list_bo_year) > 1:
            sns.kdeplot(list_bo_year,
                        color=c_nation, ec="w", lw=2, cut=0, fill=True, ax=ax, alpha=1)
            ax.set_ylabel("")

        else:
            ax.plot(list_bo_year * 2, [ax.get_ybound()[1]/2] * 2,
                        c=c_nation, lw=2, alpha=1)
            ax.set_ylim(0, np.finfo(float).eps)

    # 왼쪽 y축범위
```

```
    ymax_all = max([ax.get_ybound()[1] for ax in axes[:,0]])

    for ax, year in zip(axes[:,0], range(year0, year1+1)):
        # x, y축 범위 설정
        ax.set_xlim(0, xmax*1.05)
        ax.set_ylim(0, ymax_all)

        # xticklabels
        xticks = [x for x in ax.get_xticks() if (ax.get_xbound()[0] <= x <= ax.get_xbound()[1
])]
        if year == year1:
            ax.set_xticks(xticks)
            ax.set_xticklabels([f"{x/10000:.0f}만" if x>0 else "0" for x in xticks])
        else:
            ax.set_xticks(xticks)
            ax.set_xticklabels([])

        list_bo_year_ = df_bo_distS.query(f"openYear == {year}")["서울 관객수"].values
        if len(list_bo_year_) == 0:
            continue

        list_bo_year = list_bo_year_[0]

        # 최다 관객
        max_num = np.max(list_bo_year)
        ax.plot([max_num]*2, [-0.3*ymax_all, 0.3*ymax_all], "ko-", lw=2, mfc="#FFFF00", mew=2)

        # 해외
        if nation == '해외':
            max_movieNm = df_bo.query(f"openYear == {year}").loc[df_bo["서울 관객수"]==max_num]
.loc[df_bo["국적"]!='한국']["영화명"].values[0]
        else:
            max_movieNm = df_bo.query(f"openYear == {year}").loc[df_bo["서울 관객수"]==max_num]
.loc[df_bo["국적"]==nation]["영화명"].values[0]

        if max_num > annot_pos*xmax:
            ax.text(max_num, 0.1*ymax_all, f"{max_movieNm} ({format(max_num, ',')})  ",
                    ha="right", c="k", fontsize="small")
        else:
            ax.text(max_num, 0.1*ymax_all, f"  {max_movieNm} ({format(max_num, ',')})",
                    ha="left", c="k", fontsize="small")
        ax.set_ylim(0, )



    ### 관객 수 분포
    for ax, year in zip(axes[:,1], range(year0, year1+1)):
        ax.set_xlim(100, xmax*1.1)
        ax.set_ylim(0, ymax_all)
        ax.set_facecolor("none")
        ax.spines[["left", "top", "right"]].set_visible(False)
        ax.grid(False)

        ax.set_yticks([])
        ax.set_xscale("log")

        # xticklabels
        xticks = [x for x in ax.get_xticks() if (ax.get_xbound()[0] <= x <= ax.get_xbound()[1
])]
        xticks_minor = [x for x in ax.get_xticks(minor=True) if (ax.get_xbound()[0] < x < ax.g
et_xbound()[1])]
```

```
        if year == year1:
            ax.set_xticks(xticks)
            ax.set_xticks(xticks_minor, minor=True)
            ax.set_xticklabels([f"{x/10000:.0f}만" if x>1000 else f"{x/1000:.0f}천" if x>100 el
se f"{x/100:.0f}백" for x in xticks])
        else:
            ax.set_xticks(xticks)
            ax.set_xticks(xticks_minor, minor=True)
            ax.set_xticklabels([])

        # xgrid
        for x in xticks:
            ax.plot([x, x], [0, ymax_all*0.4], c="lightgray", zorder=-2)
        for x in xticks_minor:
            ax.plot([x, x], [0, ymax_all*0.4], c="lightgray", lw=1, zorder=-2)
        list_bo_year_ = df_bo_distS.query(f"openYear == {year}")["서울 관객수"].values
        if len(list_bo_year_) == 0:
            continue

        list_bo_year = list_bo_year_[0]
        if len(list_bo_year) > 1:
            ymax = ax.get_ybound()[1]
            list_bo_year = np.array(list_bo_year)
            ax.plot([list_bo_year.min(), np.quantile(list_bo_year, 0.1)], [0.25*ymax, 0.25*ymax
],
                    c="darkred", lw=3, solid_capstyle='butt', label="하위10% 이하", zorder=-1)
            ax.plot([np.quantile(list_bo_year, 0.1), np.median(list_bo_year)], [0.25*ymax, 0.25*
ymax],
                    c="r", lw=13, solid_capstyle='butt', label="하위10%-중간값")
            ax.plot([np.median(list_bo_year), np.quantile(list_bo_year, 0.9)], [0.25*ymax, 0.25*
ymax],
                    c="b", lw=13, solid_capstyle='butt', label="중간값-상위10%")
            ax.plot([np.quantile(list_bo_year, 0.9), list_bo_year.max()], [0.25*ymax, 0.25*ymax
],
                    c="darkblue", lw=3, solid_capstyle='butt', label="상위10% 이상", zorder=-1)
            ax.plot([np.median(list_bo_year), np.median(list_bo_year)], [0, 0.4*ymax], c="w", lw
=1)
            ax.scatter([list_bo_year.max()], [0.25*ymax], c="k", lw=2, fc="#FFFF00", ec="k", s=1
00, label="최대관객")


            ax.set_ylabel("")

        elif len(list_bo_year) == 1:
            ax.scatter([list_bo_year[0]], [0.25*ymax], c="k", lw=2, fc="#FFFF00", ec="k", s=100,
label="최대관객")

        else:
            ax.plot(list_bo_year * 2, [ax.get_ybound()[1]/2] * 2,
                        c=c_nation, lw=2, alpha=1)



    handles, labels = axes[0, 1].get_legend_handles_labels()

    if legend:
        fig.legend(handles=handles[:5], labels=labels[:5],
                loc="upper center", bbox_to_anchor=(0.5,1), frameon=True,
                fontsize="small", ncol=5)
        suptitle = " "
        fig.suptitle(suptitle,
```

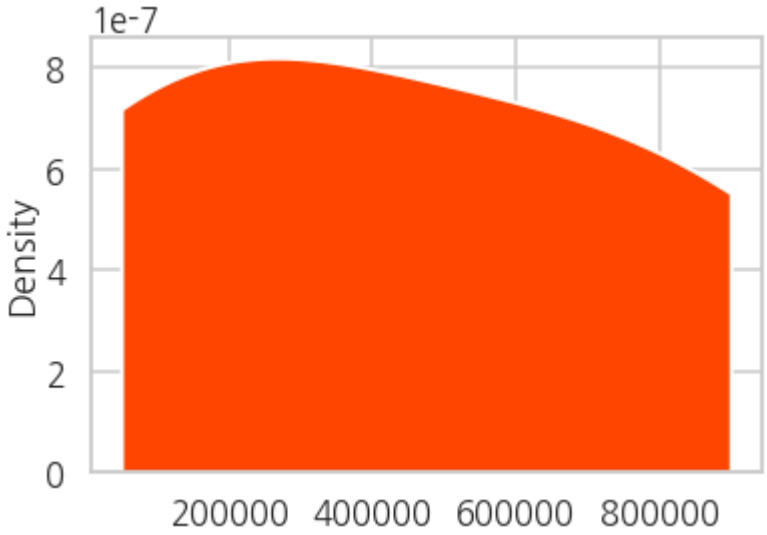
```
                fontweight="bold", color="0.4")
        fig.subplots_adjust(hspace=-0.5, wspace=0.1, left=0.06, bottom=0.15, top=0.95, right=0.9
5)
    else:
        suptitle = f"{nation} 영화 (서울 관객 기준, 명)"
        fig.suptitle(suptitle,
                fontweight="bold", color="0.4")
        fig.subplots_adjust(hspace=-0.5, wspace=0.1, left=0.06, bottom=0.05, top=0.95, right=0.9
5)

    fig.savefig(f"./images/bo_year_{year0}-{year1}_{nation}.png", dpi=200)
```

In [65]:

```
nation = '홍콩'
df_bo_distS = df_bo_genres.query(f"국적 == '{nation}'").groupby("openYear").agg(list)["서울 관
객수"].reset_index()

year = 1979
list_bo_year_ = df_bo_distS.query(f"openYear == {year}")["서울 관객수"].values
list_bo_year = list_bo_year_[0]

c_nation = c_hk
sns.kdeplot(list_bo_year,
                        color=c_nation, ec="w", lw=2, cut=0, fill=True, alpha=1)
```

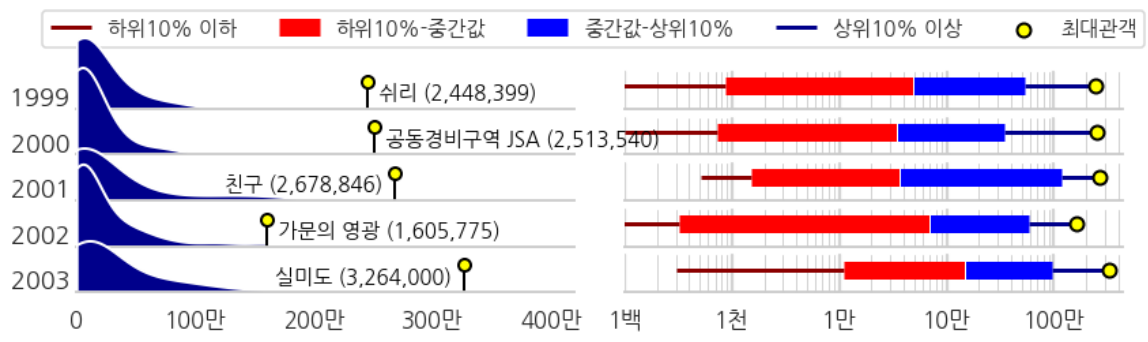
Out[65]:

<AxesSubplot:ylabel='Density'>



In [66]:

```
plot_boS_year(1999, 2003, '한국', annot_pos=0.63, legend=True, xmax=4e6)
```



In [243]:

```
def get_median(year, nation):
    df_median = df_bo_genres.query(f"국적 == '{nation}'").query(f"openYear == {year}")
    m = df_median["서울 관객수"].median()
    if df_median.loc[df_median["서울 관객수"] > m, "서울 관객수"].shape[0] > 0:
        m_idx = np.argmin(df_median.loc[df_median["서울 관객수"] > m, "서울 관객수"])
        results = df_median.loc[df_median["서울 관객수"] > m].iloc[m_idx].values
        return df_median.shape[0], results[1], results[3]
```

In [244]:

```
for y in range(1999, 2004):
    print(get_median(y, "한국"))
```

```
(42, '이재수의난', 56913)
(56, '신혼여행(身魂旅行)', 36218)
(52, '교도소월드컵', 41594)
(82, '쓰리', 69778)
(65, '...ing', 159396)
```

In [245]:

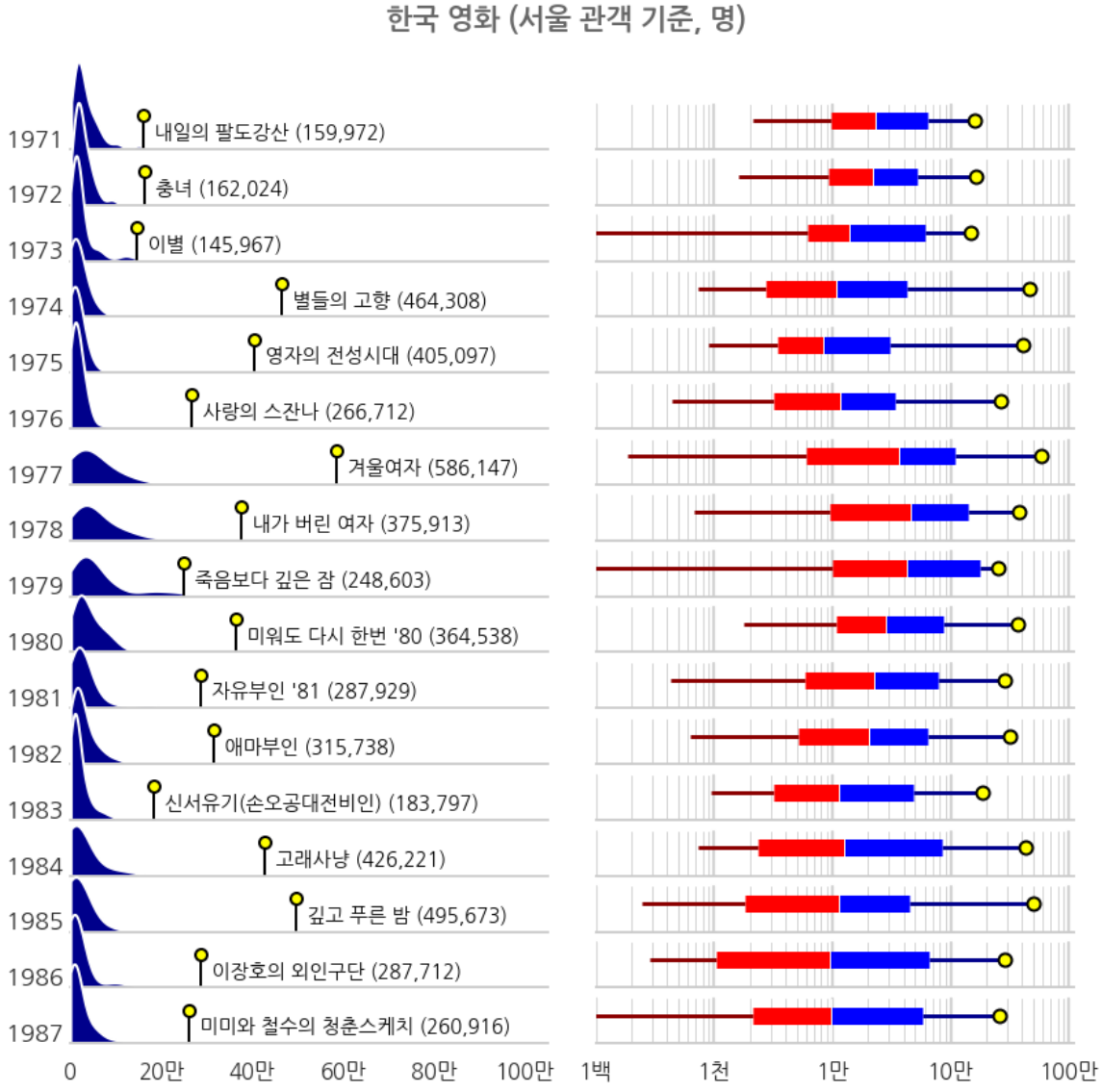
```
for y in range(1971, 1988):
    print(get_median(y, "영국"))
```

```
(7, '올리버', 109618)
(10, '렛드로즈특공대', 86978)
(3, '007제2편:위기일발', 73330)
(2, '공포의드라큐라', 151317)
(7, '돌아온드라큐라', 100399)
(2, '알프스대탈출', 199503)
(4, '샤우트', 150818)
(5, '지옥의특전대', 410049)
(2, '리벤저', 325259)
(3, '바다의늑대들', 229004)
(3, '사막의라이온', 203828)
None
None
None
(2, '킬링필드', 925994)
(3, '미션', 525630)
None
```

3.4.2. 1971-1987

In [67]:

```
plot_boS_year(1971, 1987, '한국', annot_pos=0.6, xmax=1e6)
```



In [246]:

```
for y in range(1971, 1988):
    print(get_median(y, "한국"))
```

```
(118, '미스리', 23425)
(98, '신풍협객', 21722)
(76, '동풍', 13956)
(125, '밤에도뜨는태양', 10836)
(110, '애종', 8295)
(94, '영노', 12039)
(68, '고교깡돌이', 37482)
(70, '소림천하', 46104)
(79, '혈육마방', 43614)
(78, '통천노호', 28533)
(90, '가슴깊게화끈하게', 22792)
(101, '조용한방', 20598)
(99, '비련', 12025)
(73, '대학괴짜들', 12791)
(88, '밤을벗기는독장미', 11367)
(86, '화랭이', 9630)
(88, '먼여행긴터널', 9729)
```

In [68]:

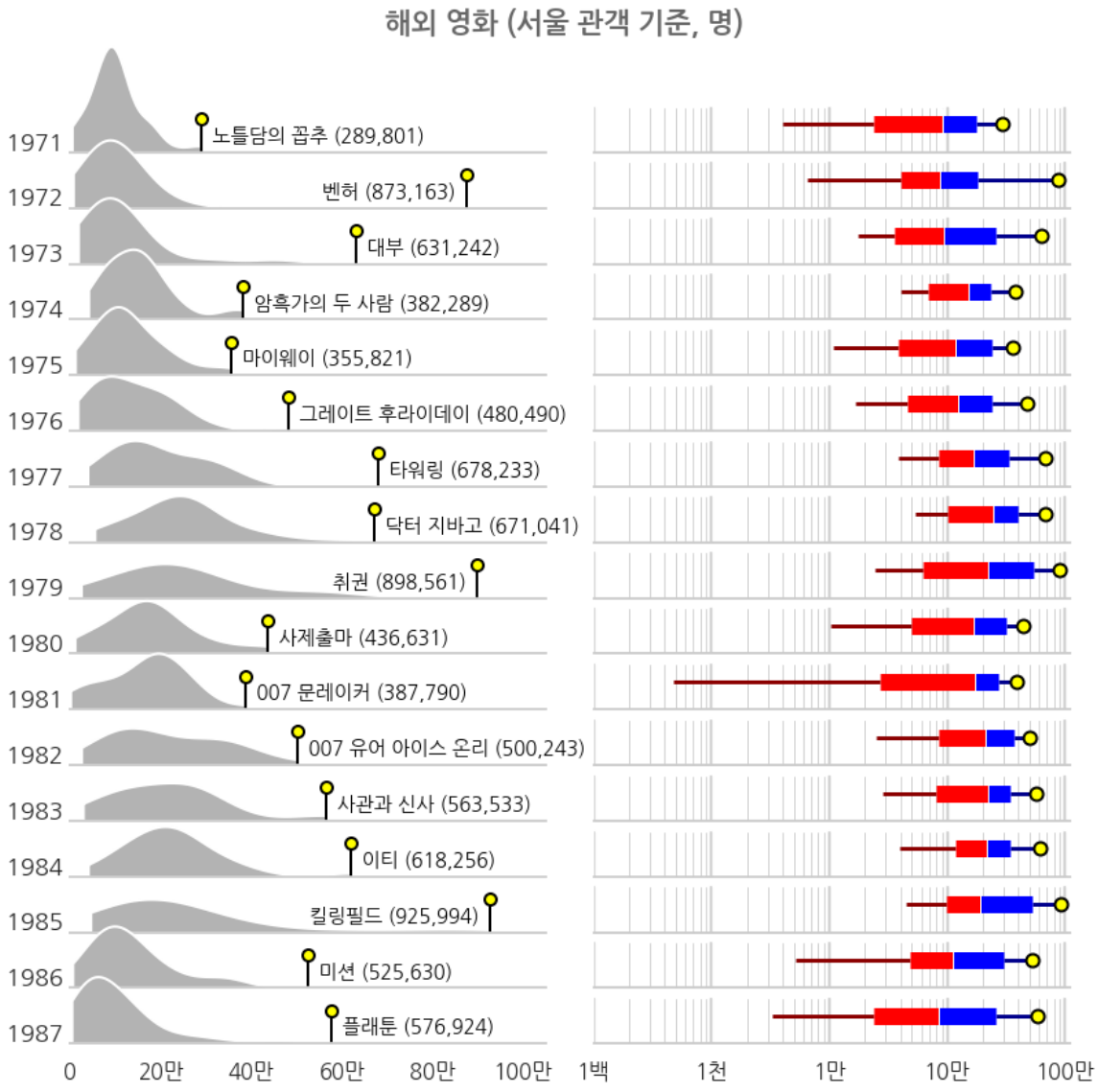
```
df_bo_genres.query("국적 == '한국'").query("openYear <= 1987").loc[df_bo_genres["서울 관객수"] < 100]
```

Out[68]:

	movieCd	영화명	개봉일	서울관객수	전국관객수	국적	openYear	G_SF	G_가족	G_공연	G_공포	G_기타	G_다큐멘터리	G_드라마	G_멜로/로맨스	G
17273	19730102	언제나님과함께	1973-06-06	20	0	한국	1973	0	0	0	0	0	0	0	1	
17591	19790052	낙조	1979-05-13	84	0	한국	1979	0	0	0	0	0	0	1	1	
22826	19870030	잊을수없는순간	1987-06-01	80	0	한국	1987	0	0	0	0	1	0	0	0	

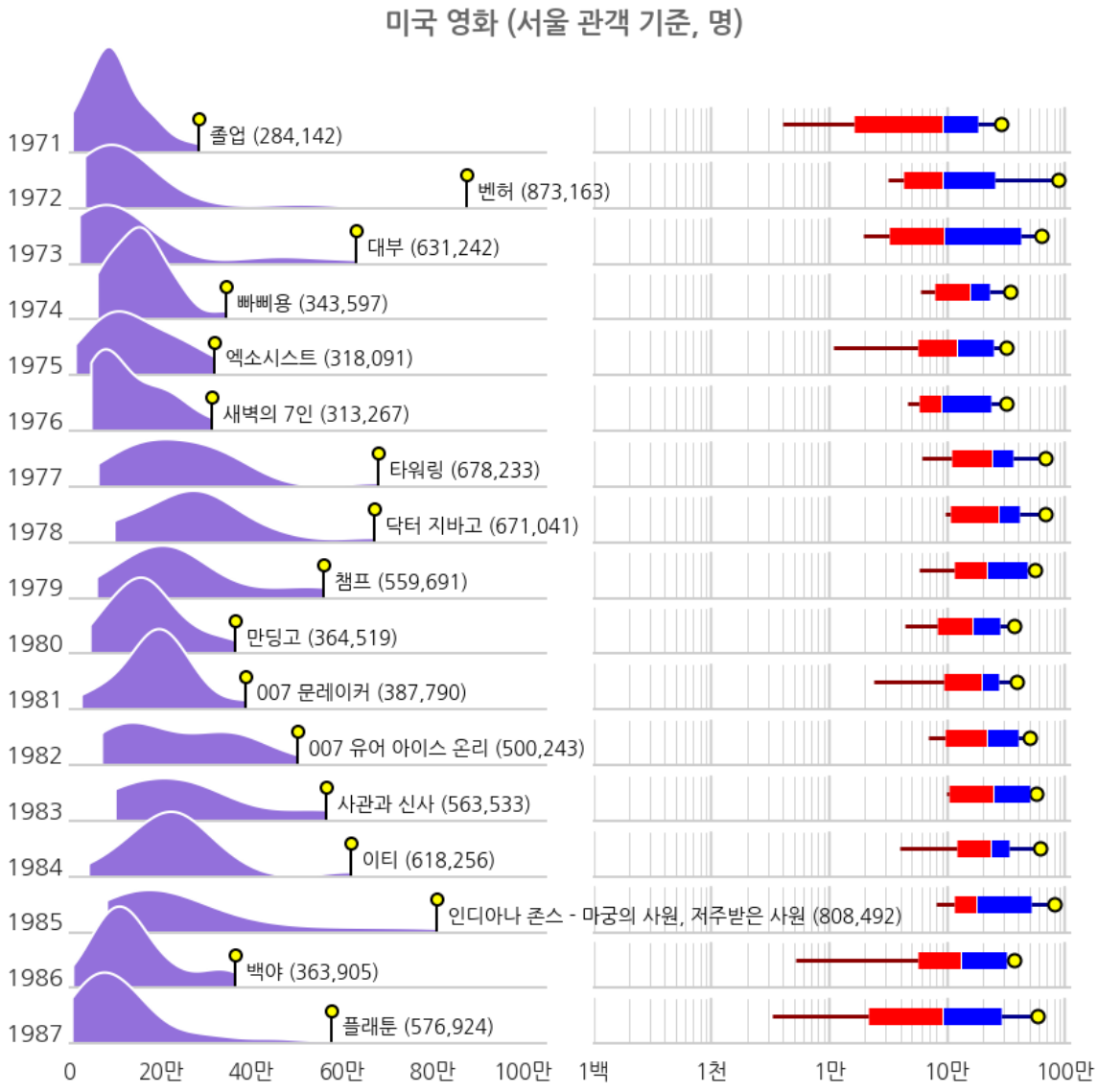
In [69]:

```
plot_boS_year(1971, 1987, '해외', annot_pos=0.7, xmax=1e6)
```



In [70]:

```
plot_boS_year(1971, 1987, '미국', annot_pos=0.9, xmax=1e6)
```



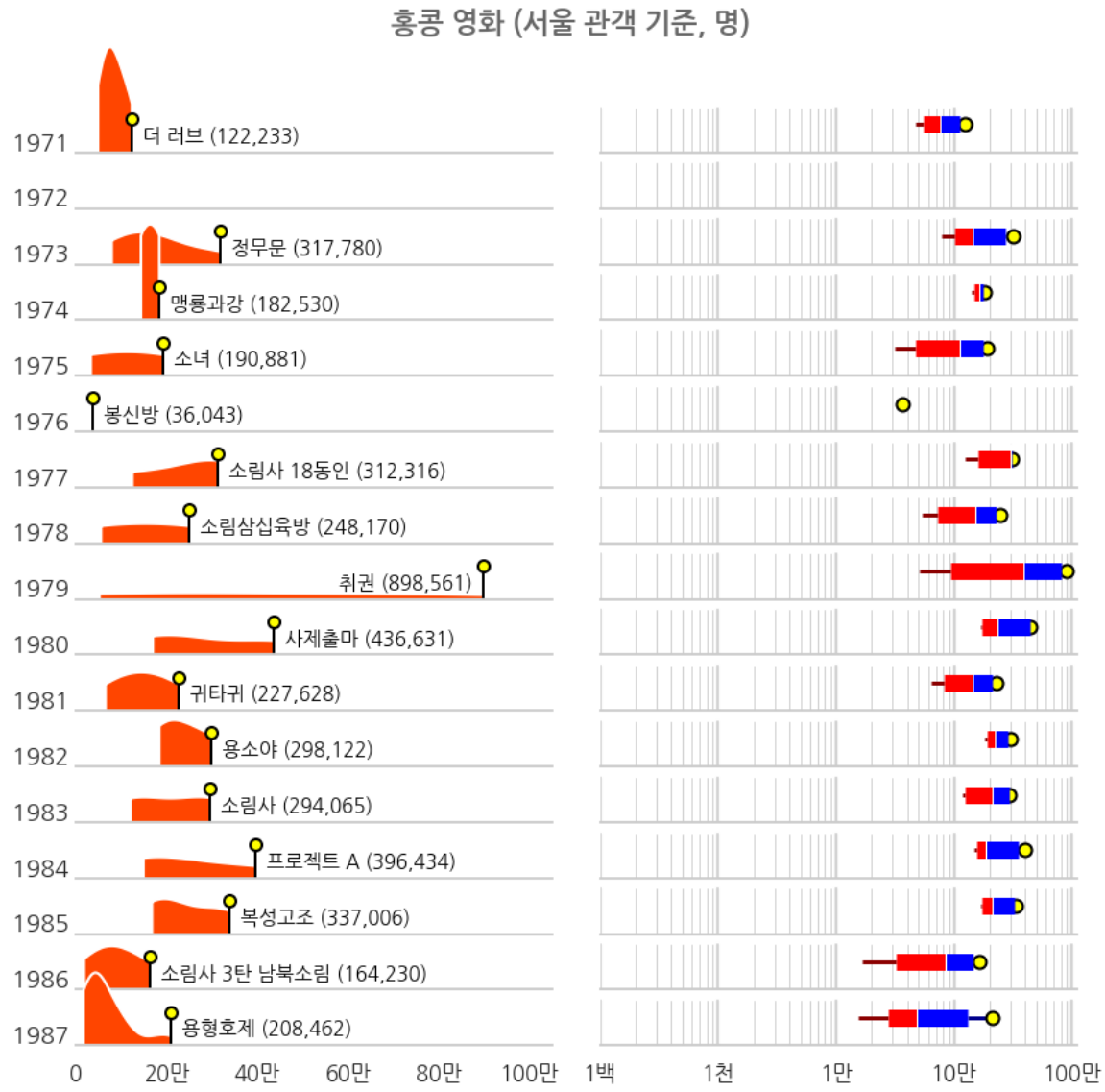
In [247]:

```
for y in range(1971, 1988):
    print(get_median(y, "미국"))
```

```
(35, '맛슈', 92146)
(35, '딸라', 90944)
(22, '애심', 93727)
(21, '삼손과데릴라', 158373)
(12, '스콜피오', 129380)
(17, '오뎃사화일', 118545)
(22, '실버스트릭', 238855)
(17, '사망유희', 281591)
(19, '죠스2', 220598)
(19, '예스터데이', 166843)
(16, '500만달러의모험', 201982)
(19, '스워드', 218606)
(14, '투씨', 257818)
(21, '사하라', 234181)
(19, '폴리스아카데미', 203681)
(34, '볼레로', 131133)
(49, '록키4', 92011)
```

In [71]:

```
plot_boS_year(1971, 1987, '홍콩', annot_pos=0.7, xmax=1e6)
```



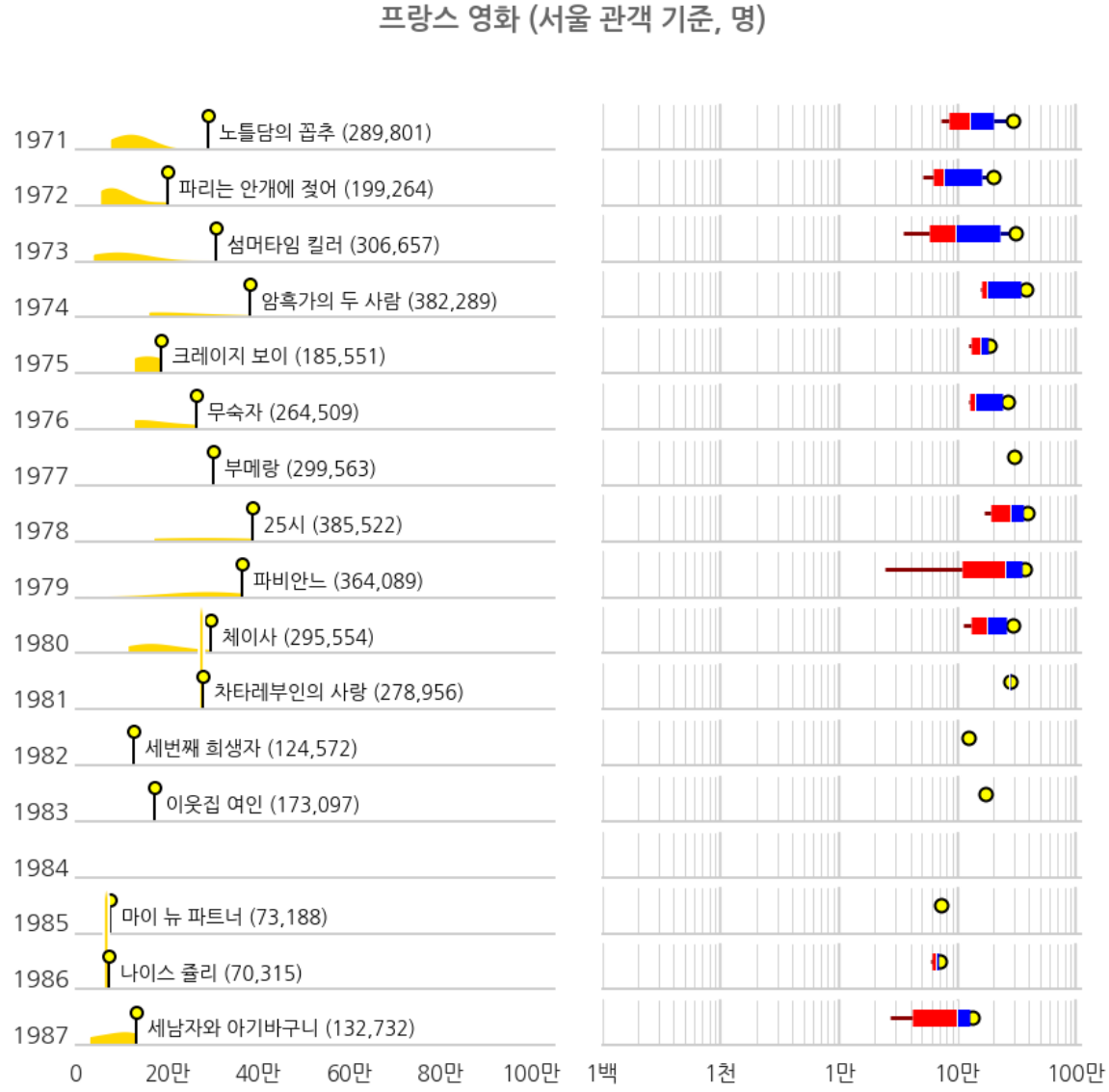
In [248]:

```
for y in range(1971, 1988):
    print(get_median(y, "홍콩"))
```

```
(7, '권격', 90610)
None
(5, '당산대형', 209551)
(2, '맹룡과강', 182530)
(2, '소녀', 190881)
None
(3, '소림사18동인', 312316)
(2, '소림삼십육방', 248170)
(4, '사형도수', 576953)
(5, '소권괴초', 436545)
(4, '남북취권', 165563)
(3, '용소야', 298122)
(4, '오복성', 285579)
(3, '프로젝트A', 396434)
(5, '괘찬차', 307751)
(4, '예스마담-황가사저', 100155)
(17, '파우', 49355)
```

In [72]:

```
plot_boS_year(1971, 1987, '프랑스', annot_pos=0.7, xmax=1e6)
```



In [249]:

```
for y in range(1971, 1988):
    print(get_median(y, "프랑스"))
```

```
(8, '밤의불청객', 131612)
(6, '대도적', 77585)
(6, '연인들의장소', 102030)
(3, '암흑가의두사람', 382289)
(2, '크레이지보이', 185551)
(3, '무숙자', 264509)
None
(2, '25시', 385522)
(5, '루지탕', 318962)
(4, '후릭크', 177222)
(2, '차타레부인의사랑', 278956)
None
None
None
None
(2, '나이스쥴리', 70315)
(3, '세남자와아기바구니', 132732)
```

In [73]:

```
# 킬링필드 국적 확인: 영국
df_bo.query("영화명 == '킬링필드'")
```

Out[73]:

	순번	영화명	감독	제작사	수입사	배급사	개봉일	영화유형	영화형태	국적	전국스크린수	전국매출액	전국관객수	서울매출액	서울관객수	장르	등급	영화구분
22181	22182	킬링필드	롤랑조페	워너브러더스픽쳐스	NaN	NaN	1985-06-01	개봉영화	장편	영국	0	0.0	0	0.0	925994	드라마	연소자관람가	일반영화

In [74]:

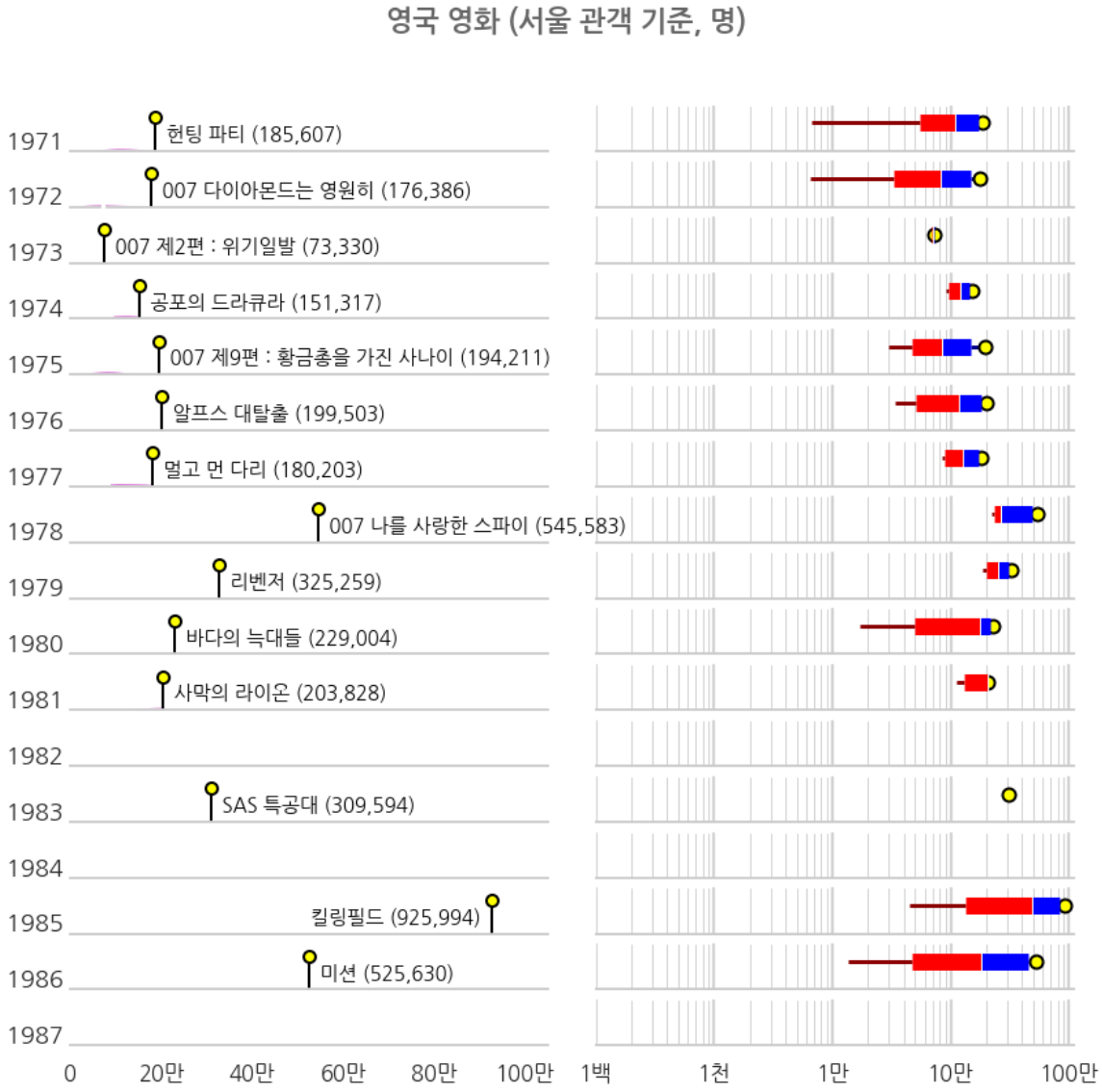
```
# 킬링필드 국적 확인: 영국
df_bo.query("영화명 == '미션'")
```

Out[74]:

	순번	영화명	감독	제작사	수입사	배급사	개봉일	영화유형	영화형태	국적	전국스크린수	전국매출액	전국관객수	서울매출액	서울관객수	장르	등급
18136	18137	미션	두기봉	밀키웨이미지	오성미디어(주)	NaN	2000-04-22	개봉영화	장편	홍콩	0	0.0	0	0.0	5376	액션	18세관람가
18137	18138	미션	롤랑조페	NaN	주식회사더쿱,(주)피터팬픽쳐스	주식회사더쿱,(주)피터팬픽쳐스	1986-12-24	개봉영화	장편	영국	0	0.0	0	0.0	525630	드라마	12세이상관람가,국민학생관람불가,15세이상관람가

In [75]:

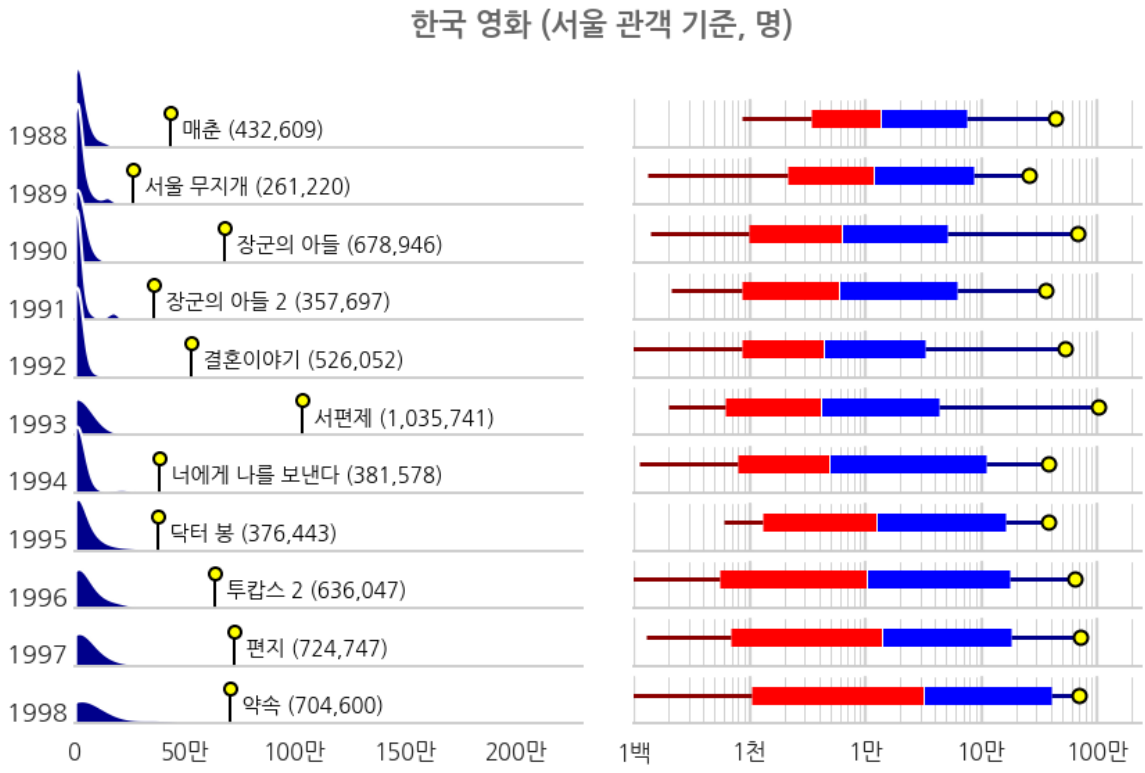
```
plot_boS_year(1971, 1987, '영국', annot_pos=0.7, xmax=1e6)
```



3.4.3. 1988-1998

In [76]:

```
plot_boS_year(1988, 1998, '한국', annot_pos=0.6, xmax=2.2e6)
```



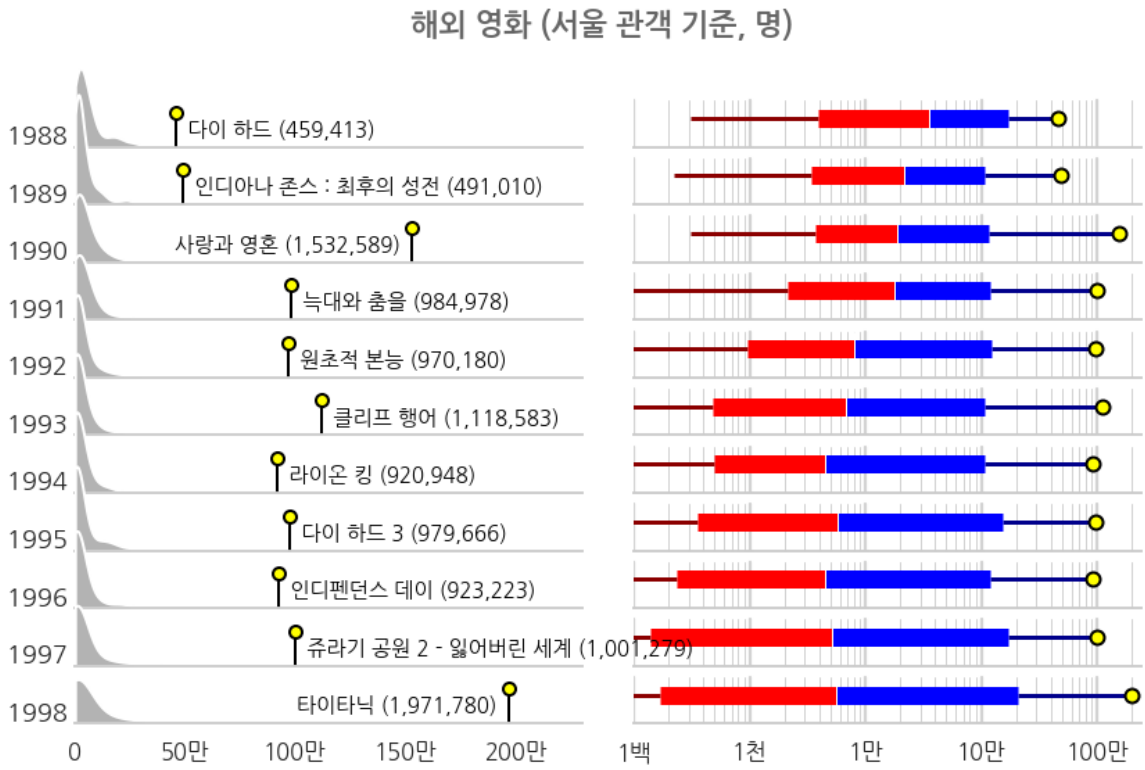
In [250]:

```
for y in range(1988, 1998):
    print(get_median(y, "한국"))
```

```
(80, '후궁별곡', 13468)
(85, '미쓰코뿔소미스터코란도', 11973)
(102, '늪속의여촌장', 6381)
(115, '사랑은지금부터시작이야', 6029)
(89, '탄드라부인', 5273)
(70, '오렌지나라', 4246)
(65, '티라노의발톱', 5145)
(59, '48+1', 13524)
(55, '1996뽕', 14971)
(60, '마지막방위', 14223)
```

In [77]:

```
plot_boS_year(1988, 1998, '해외', annot_pos=0.6, xmax=2.2e6)
```

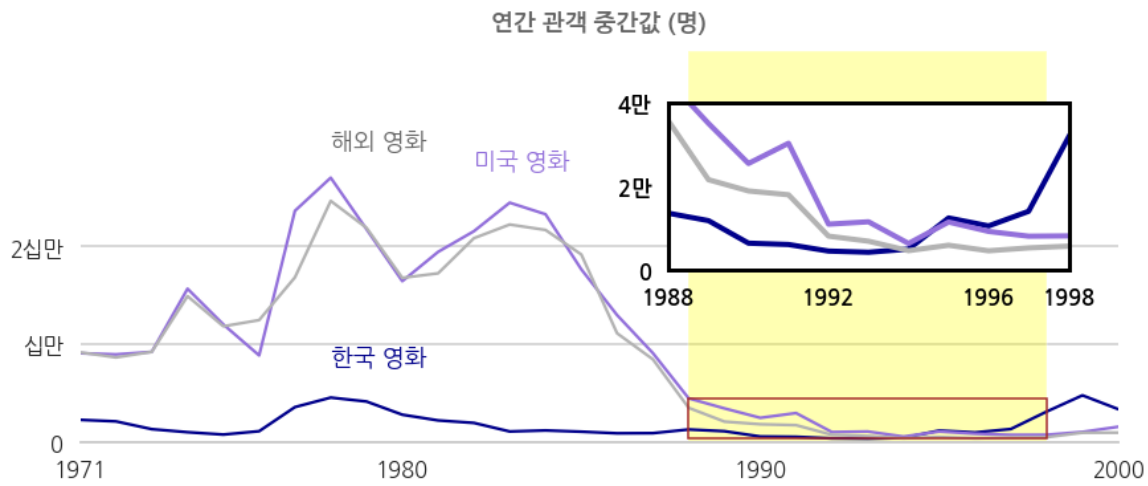


In [78]:

```
nation = "한국"
median_kr = df_bo_genres.query(f"국적 == '한국'").groupby("openYear").agg(np.median)["서울 관객수"].reset_index()
median_us = df_bo_genres.query(f"국적 == '미국'").groupby("openYear").agg(np.median)["서울 관객수"].reset_index()
median_nkr = df_bo_genres.query(f"국적 != '한국'").groupby("openYear").agg(np.median)["서울 관객수"].reset_index()
```

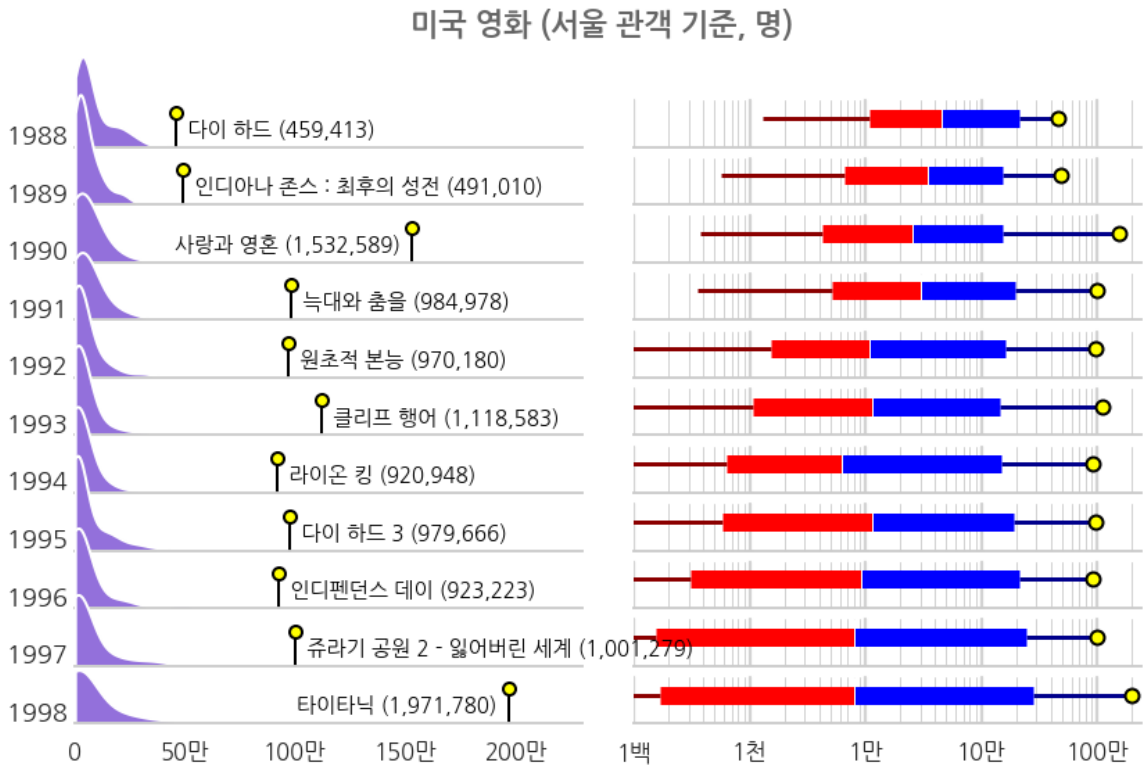
In [79]:

```
from matplotlib.patches import Rectangle

fig, ax = plt.subplots(figsize=(12, 5), constrained_layout=True)

ax.plot(median_kr["openYear"], median_kr["서울 관객수"], c=c_kr)
ax.plot(median_us["openYear"], median_us["서울 관객수"], c=c_us)
ax.plot(median_nkr["openYear"], median_nkr["서울 관객수"], c=c_etc)
ax.set_xlim(1971, 2000)
ax.set_title("연간 관객 중간값 (명)", fontdict=font_title, pad=16)
ax.spines[["top", "left", "right"]].set_visible(False)
ax.grid(axis="x")
xticks = [1971, 1980, 1990, 2000]
ax.set_xticks(xticks)
ax.set_xticklabels(xticks)
ax.set_ylim(0, 4e5)
ax.set_yticks([0, 1e5, 2e5])
ax.set_yticklabels([0, "십만", "2십만"])

rec = Rectangle([1988, 5e3], 10, 4e4, fc="none", ec="brown", zorder=10)
ax.add_artist(rec)

ax_big = fig.add_axes([0.58, 0.45, 0.35, 0.35])
ax_big.plot(median_kr["openYear"], median_kr["서울 관객수"], lw=4, c=c_kr)
ax_big.plot(median_us["openYear"], median_us["서울 관객수"], lw=4, c=c_us)
ax_big.plot(median_nkr["openYear"], median_nkr["서울 관객수"], lw=4, c=c_etc)
ax_big.set_xlim(1988, 1998)
ax_big.set_ylim(0,40000)
ax_big.grid(False)
ax_big.spines[["top", "left", "right", "bottom"]].set_edgecolor("k")
ax_big.spines[["top", "left", "right", "bottom"]].set_linewidth(3)
ax_big.set_xticks([1988, 1992, 1996, 1998])
ax_big.set_xticklabels([1988, 1992, 1996, 1998], color="k", fontweight="bold")
ax_big.set_yticks([0, 2e4, 4e4])
ax_big.set_yticklabels([0, "2만", "4만"], color="k", fontweight="bold")

ax.axvspan(1988, 1998, fc="#FFFF00", alpha=0.3)
ax.text(1978, 8e4, "한국 영화", c=c_kr)
ax.text(1982, 2.8e5, "미국 영화", c=c_us)
ax.text(1978, 3e5, "해외 영화", c="0.4")

fig.savefig("./images/bo_median_1971_2000.png", dpi=200)
```



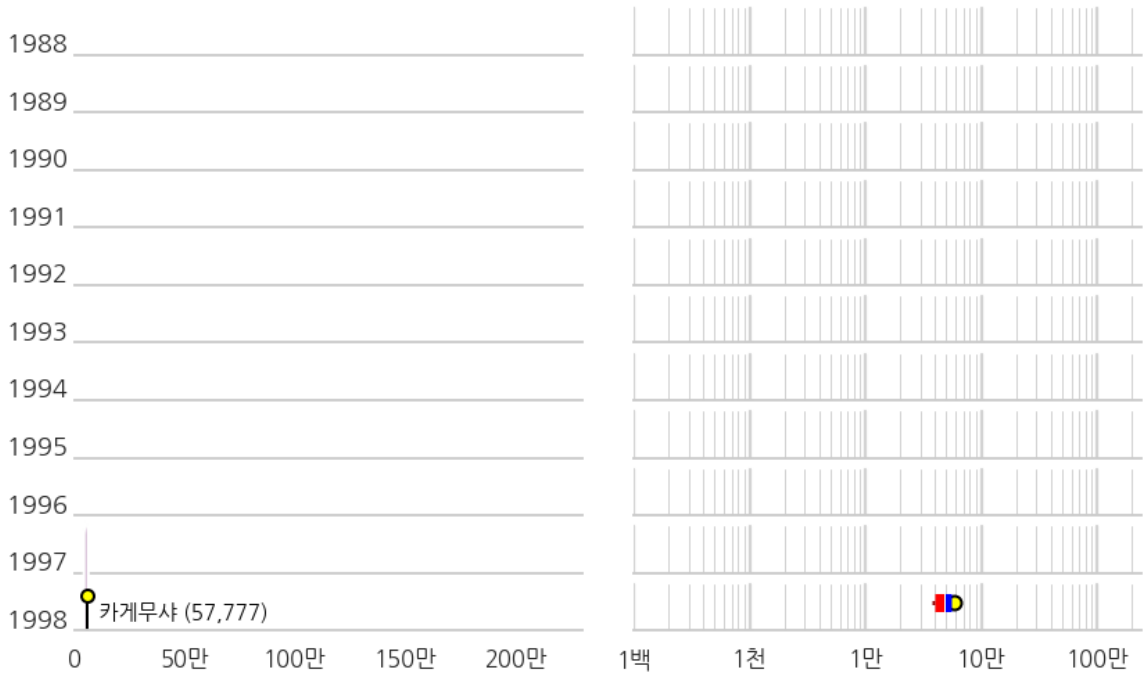
In [80]:

```
plot_boS_year(1988, 1998, '미국', annot_pos=0.6, xmax=2.2e6)
```



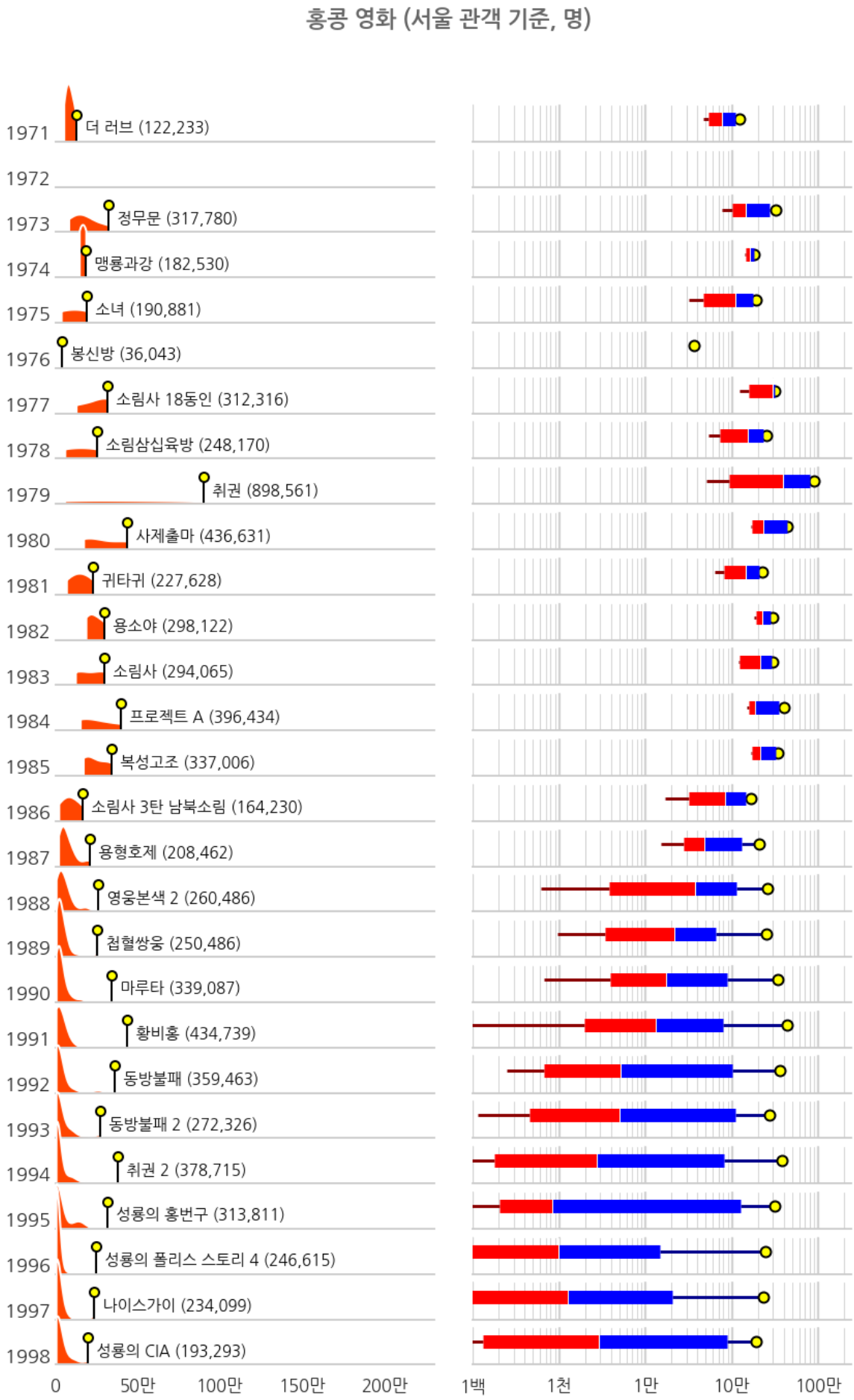
In [81]:

```
plot_boS_year(1988, 1998, '일본', annot_pos=0.6, xmax=2.2e6)
```



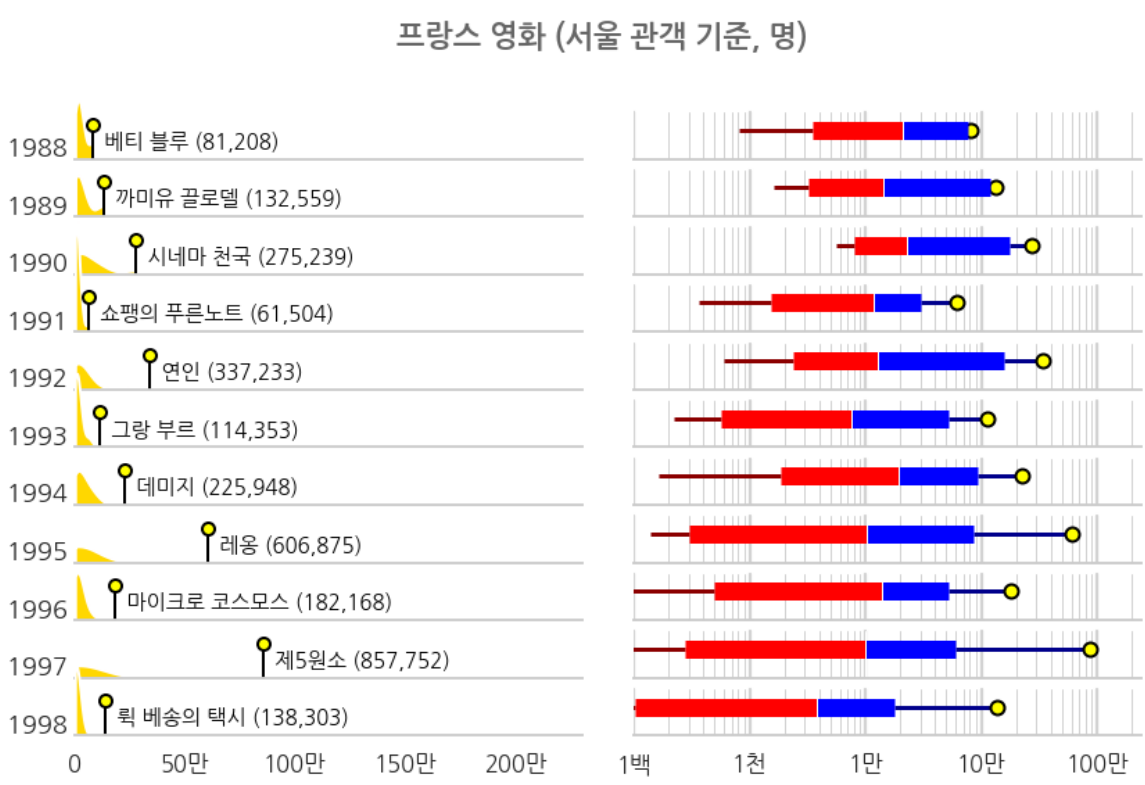
In [82]:

```
plot_boS_year(1971, 1998, '홍콩', annot_pos=0.6, xmax=2.2e6)
```


홍콩 영화 (서울 관객 기준, 명)
1971 더 러브 (122,233)
1972
1973 정무문 (317,780)
1974 맹룡과강 (182,530)
1975 소녀 (190,881)
1976 봉신방 (36,043)
1977 소림사 18동인 (312,316)
1978 소림삼십육방 (248,170)
1979 취권 (898,561)
1980 사제출마 (436,631)
1981 귀타귀 (227,628)
1982 용소야 (298,122)
1983 소림사 (294,065)
1984 프로젝트 A (396,434)
1985 복성고조 (337,006)
1986 소림사 3탄 남북소림 (164,230)
1987 용형호제 (208,462)
1988 영웅본색 2 (260,486)
1989 첩혈쌍웅 (250,486)
1990 마루타 (339,087)
1991 황비홍 (434,739)
1992 동방불패 (359,463)
1993 동방불패 2 (272,326)
1994 취권 2 (378,715)
1995 성룡의 홍번구 (313,811)
1996 성룡의 폴리스 스토리 4 (246,615)
1997 나이스가이 (234,099)
1998 성룡의 CIA (193,293)
0 50만 100만 150만 200만
1백 1천 1만 10만 100만

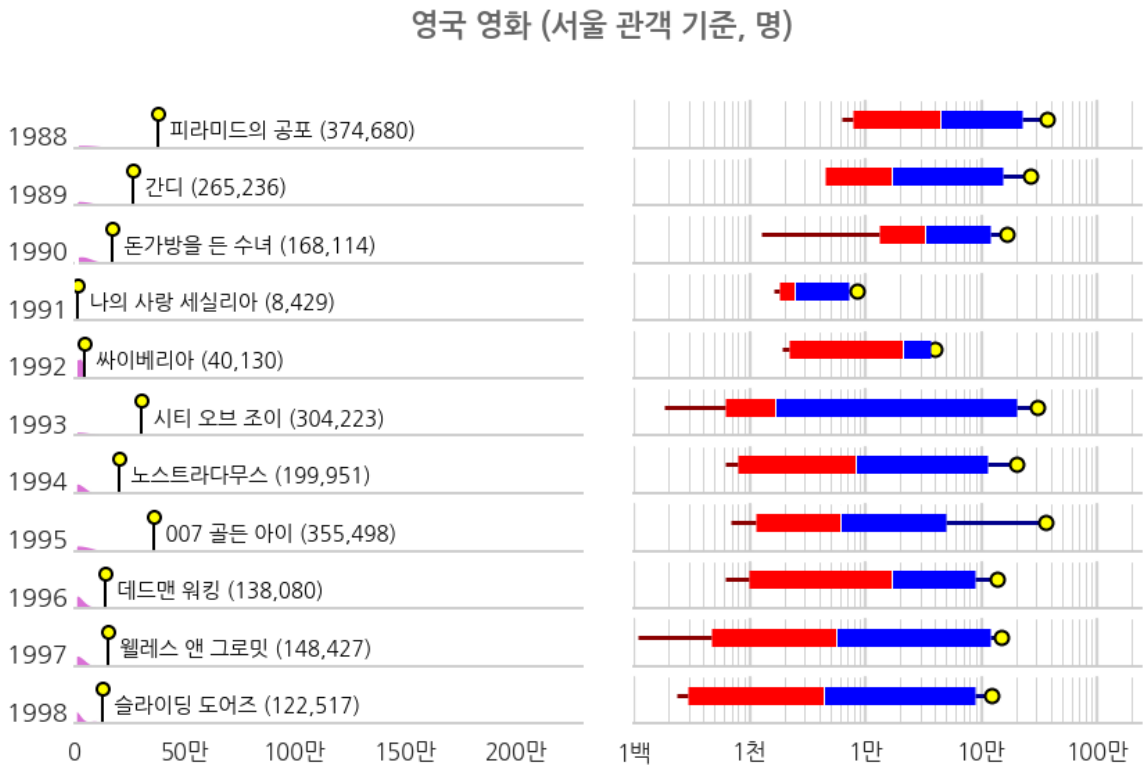
In [83]:

```
plot_boS_year(1988, 1998, '프랑스', annot_pos=0.6, xmax=2.2e6)
```



In [84]:

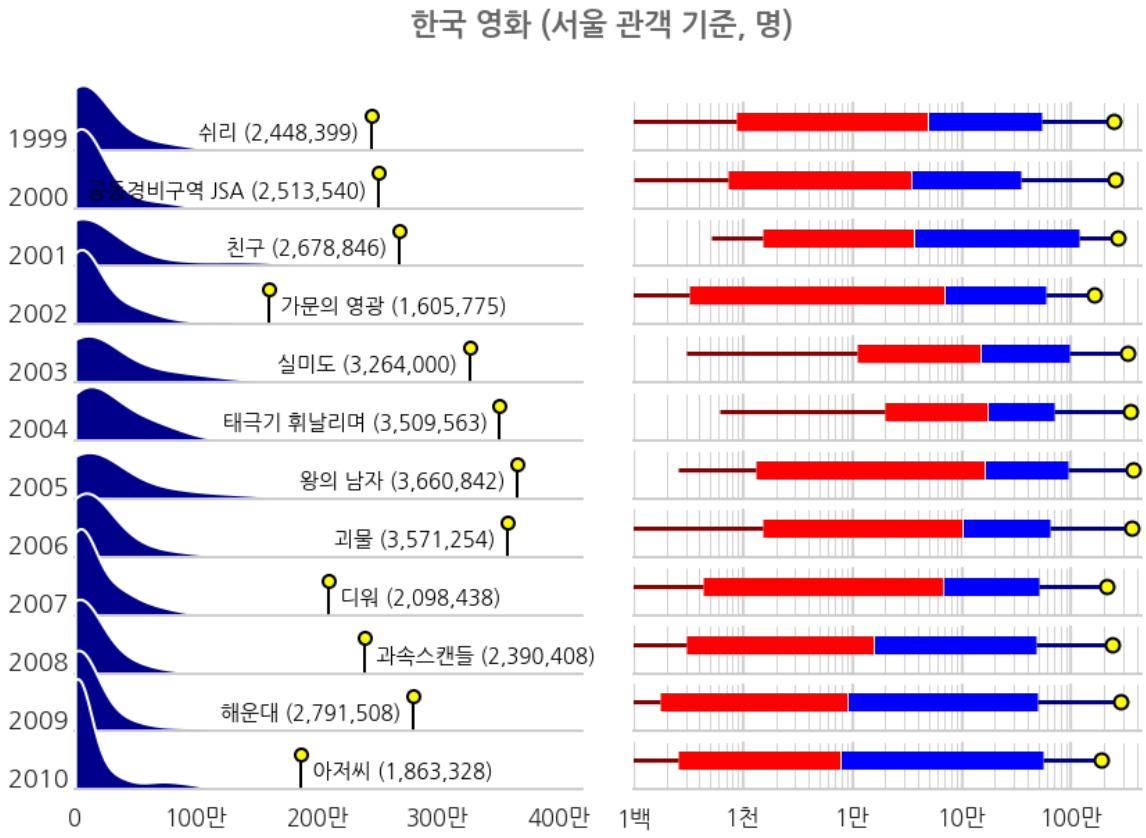
```
plot_boS_year(1988, 1998, '영국', annot_pos=0.6, xmax=2.2e6)
```



3.4.3. 1999-2010

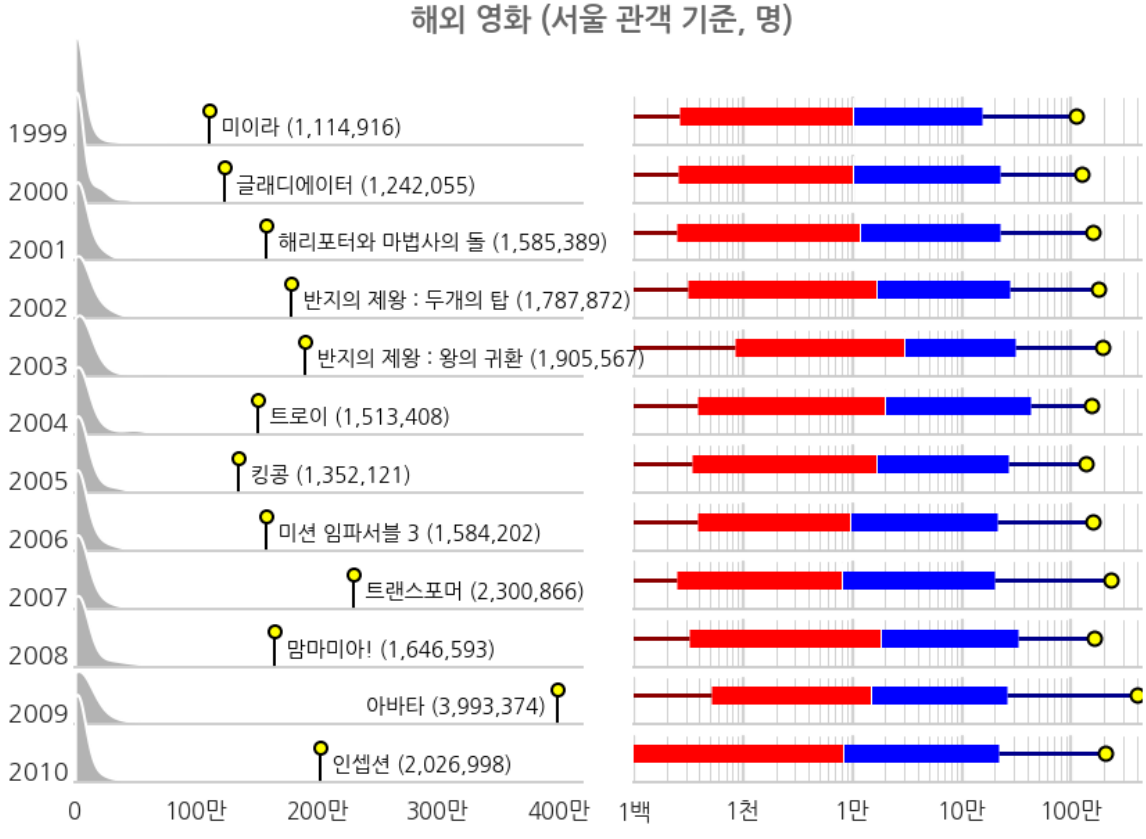
In [85]:

```
plot_boS_year(1999, 2010, '한국', annot_pos=0.6, xmax=4e6)
```



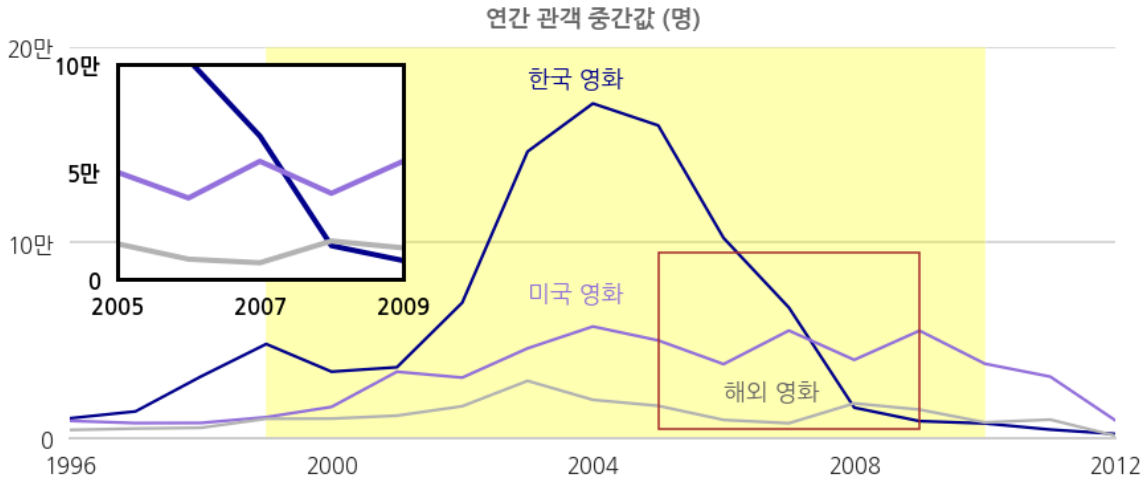
In [86]:

```
plot_boS_year(1999, 2010, '해외', annot_pos=0.6, xmax=4e6)
```



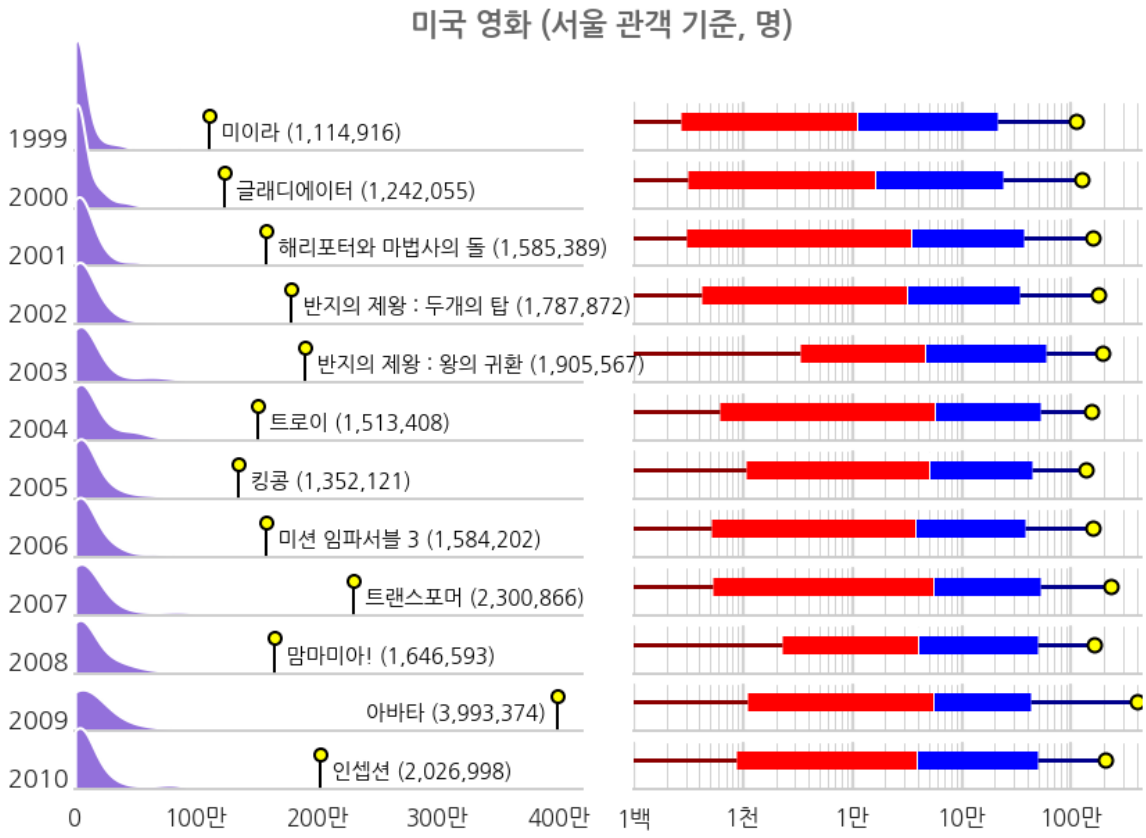
In [87]:

```
fig, ax = plt.subplots(figsize=(12, 5), constrained_layout=True)

ax.plot(median_kr["openYear"], median_kr["서울 관객수"], c=c_kr)
ax.plot(median_us["openYear"], median_us["서울 관객수"], c=c_us)
ax.plot(median_nkr["openYear"], median_nkr["서울 관객수"], c=c_etc)
ax.set_xlim(1996, 2012)
ax.set_title("연간 관객 중간값 (명)", fontdict=font_title, pad=16)
ax.spines[["top", "left", "right"]].set_visible(False)
ax.grid(axis="x")
xticks = [1996, 2000, 2004, 2008, 2012]
ax.set_xticks(xticks)
ax.set_xticklabels(xticks)
ax.set_ylim(0, 2e5)
ax.set_yticks([0, 1e5, 2e5])
ax.set_yticklabels([0, "10만", "20만"])

rec = Rectangle([2005, 5e3], 4, 9e4, fc="none", ec="brown", zorder=10)
ax.add_artist(rec)

ax_big = fig.add_axes([0.1, 0.42, 0.25, 0.45])
ax_big.plot(median_kr["openYear"], median_kr["서울 관객수"], lw=4, c=c_kr)
ax_big.plot(median_us["openYear"], median_us["서울 관객수"], lw=4, c=c_us)
ax_big.plot(median_nkr["openYear"], median_nkr["서울 관객수"], lw=4, c=c_etc)
ax_big.set_xlim(2005, 2009)
ax_big.set_ylim(0,100000)
ax_big.grid(False)
ax_big.spines[["top", "left", "right", "bottom"]].set_edgecolor("k")
ax_big.spines[["top", "left", "right", "bottom"]].set_linewidth(3)
ax_big.set_xticks([2005, 2007, 2009])
ax_big.set_xticklabels([2005, 2007, 2009], color="k", fontweight="bold")
ax_big.set_yticks([0, 5e4, 1e5])
ax_big.set_yticklabels([0, "5만", "10만"], color="k", fontweight="bold")

ax.axvspan(1999, 2010, fc="#FFFF00", alpha=0.3)
ax.text(2003, 1.8e5, "한국 영화", c=c_kr)
ax.text(2003, 0.7e5, "미국 영화", c=c_us)
ax.text(2006, .2e5, "해외 영화", c="0.4")

fig.savefig("./images/bo_median_1996_2012.png", dpi=200)
```



In [88]:

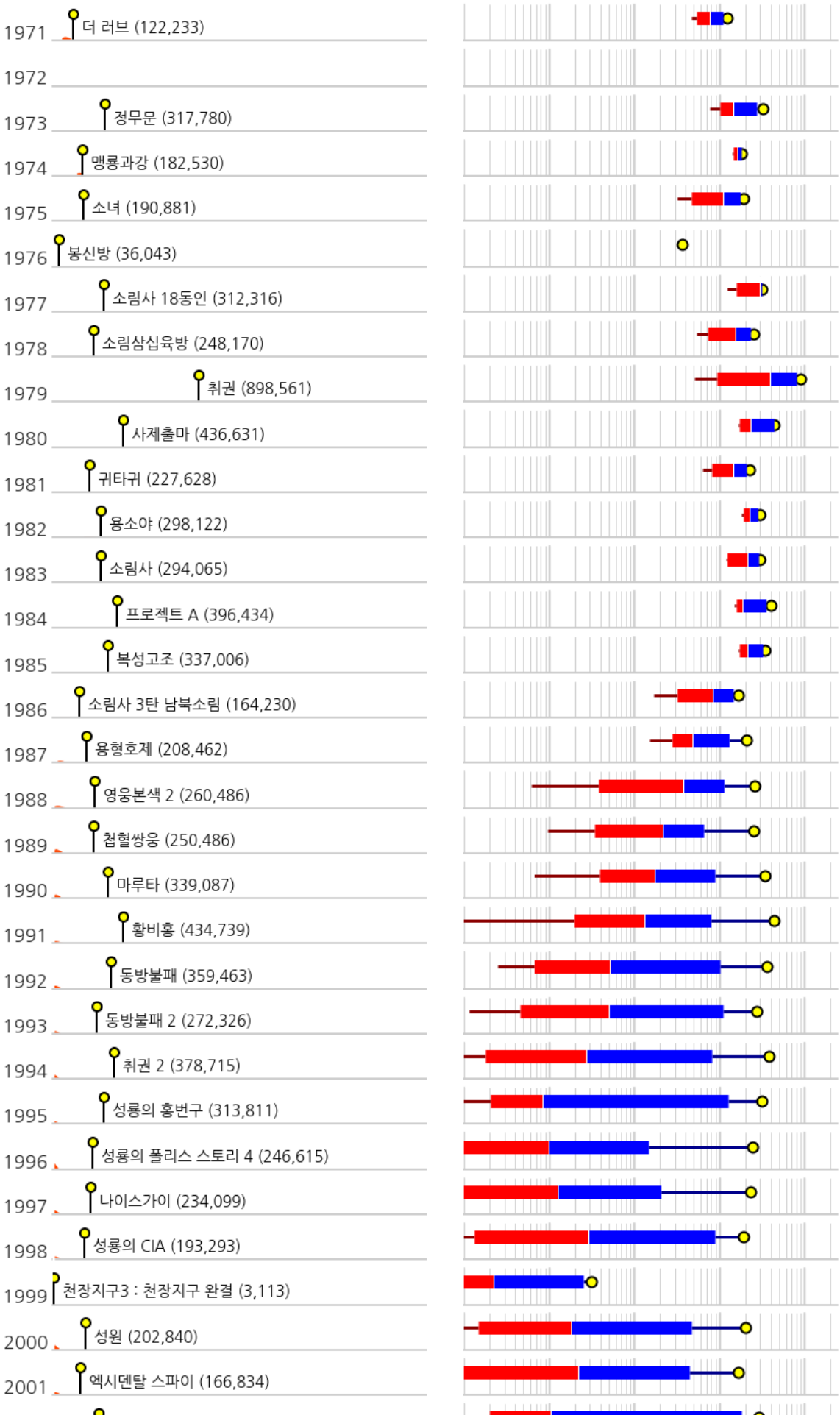
```
plot_boS_year(1999, 2010, '미국', annot_pos=0.6, xmax=4e6)
```

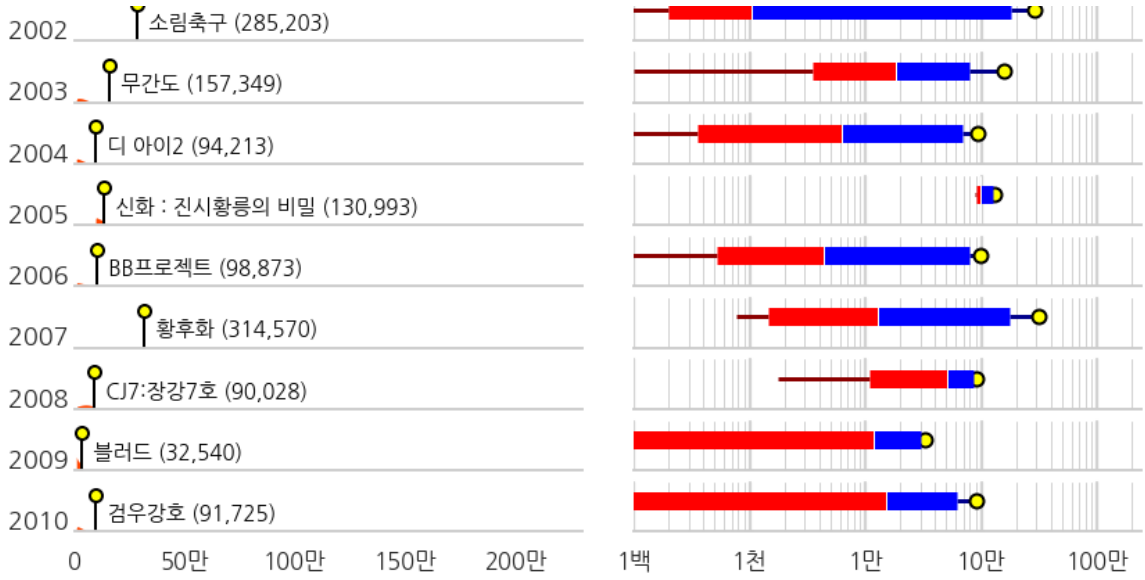


In [253]:

```
plot_boS_year(1971, 2010, '홍콩', annot_pos=0.6, xmax=2.2e6)
```

홍콩 영화 (서울 관객 기준, 명)





In [89]:

```
df_bo.query("영화명 == '아저씨'")
```

Out[89]:

	순번	영화명	감독	제작사	수입사	배급사	개봉일	영화유형	영화형태	국적	전국스크린수	전국 매출액	전국 관객수	서울 매출액
71	72	아저씨	이정범	오퍼스픽쳐스(유)	NaN	CJ ENM	2010-08-04	개봉영화	장편	한국	478	4.787079e+10	6282774	1.482086e+1

In [90]:

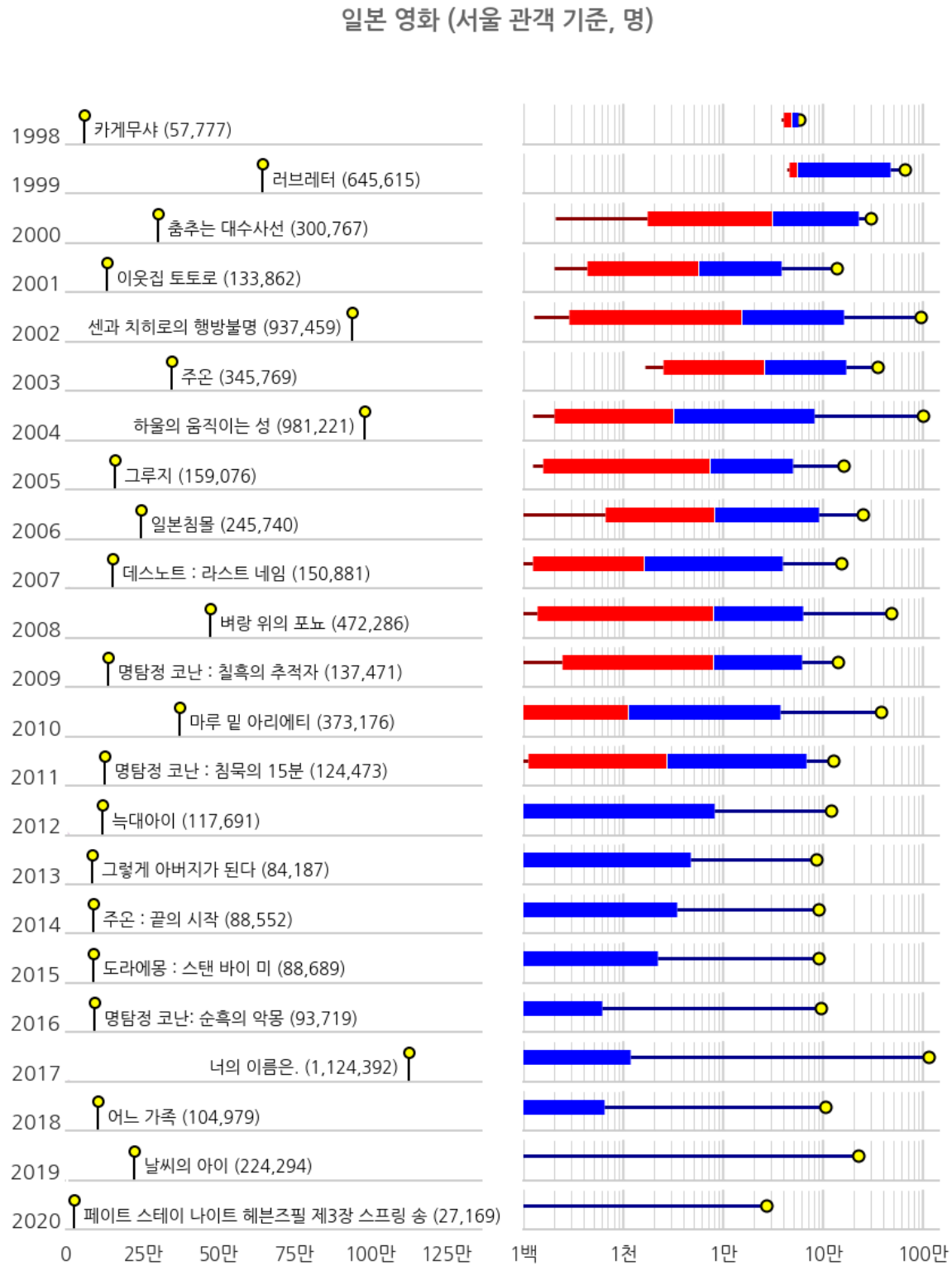
```
df_bo.query("영화명 == '인셉션'")
```

Out[90]:

	순번	영화명	감독	제작사	수입사	배급사	개봉일	영화유형	영화형태	국적	전국스크린수	전국 매출액	전국 관객수	서울 매출액
79	80	인셉션	크리스토퍼 놀란	워너브러더스픽쳐스,레젠데리픽쳐스	(주)엔케이컨텐츠,워너브러더스코리아(주)	(주)디스테이션,워너브러더스코리아(주)	2010-07-21	개봉영화	장편	미국	528	4.410655e+10	5926948	1.581048e+10

In [91]:

```
plot_boS_year(1998, 2020, '일본', annot_pos=0.6, xmax=1.3e6)
```



3.4.3.1. 일본영화

In [92]:

```
# 일본영화 장르 변화 확인 : 총 4279편
df_bo_genres_jp = df_bo_genres.query("국적 == '일본'")
print(df_bo_genres_jp.shape)
df_bo_genres_jp.head()
```

(4279, 29)

Out[92]:

	movieCd	영화명	개봉일	서울관객수	전국관객수	국적	openYear	G_SF	G_가족	G_공연	G_공포	G_기타	G_다큐멘터리	G_드라마	G_멜로/로맨스
3	20030039	링 0-버스데이	2003-04-11	3193	5806	일본	2003	0	0	0	1	0	0	0	0
5	19990317	링	1999-12-11	56983	0	일본	1999	0	0	0	1	0	0	0	0
15	20139083	언어의 정원	2013-08-14	28007	51337	일본	2013	0	0	0	0	0	0	0	0
16	20070202	초속 5 센티미터	2007-06-21	21448	21448	일본	2007	0	0	0	0	0	0	1	1
17	20111116	별을 쫓는 아이	2011-08-25	23225	52159	일본	2011	0	0	0	0	0	0	0	0

In [93]:

```
# 서울 관객 100명 이하 영화: 총 3319편
df_bo_genres_jp.loc[df_bo_genres_jp["서울 관객수"] < 100]
```

Out[93]:

	movieCd	영화명	개봉일	서울관객수	전국관객수	국적	openYear	G_SF	G_가족	G_공연	G_공포	G_기타	G_다큐멘터리	G_드라마	G_멜로/로맨스
186	20202852	욕정의 며느리 3	2020-12-10	60	60	일본	2020	0	0	0	0	0	0	1	1
190	20178342	치하야후루 상편	2018-02-28	3	36	일본	2018	0	0	0	0	0	0	1	0
191	20178343	치하야후루 하편	2018-02-28	3	36	일본	2018	0	0	0	0	0	0	1	0
215	20100354	단지 처, 오후의 정사	2010-06-24	59	60	일본	2010	0	0	0	0	0	0	0	0
229	20100355	뒤에서 앞에서	2010-06-24	56	57	일본	2010	0	0	0	0	0	0	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
18904	20100714	노트: 아내와 남교수의 금지된 사랑	2010-07-13	2	2	일본	2010	0	0	0	0	0	0	1	0

	movieCd	영화명	개봉일	서울관객수	전국관객수	국적	openYear	G_SF	G_가족	G_공연	G_공포	G_기타	G_다큐멘터리	G_드라마	G_멜로/로맨스
18905	20100702	요녀전설: 사악한 욕망	2010-07-12	2	2	일본	2010	0	0	0	1	0	0	0	0
18907	20100588	아내그리고밀애	2010-07-09	1	1	일본	2010	0	0	0	0	0	0	1	0
21490	20080150	흑야	2008-03-13	11	22	일본	2008	0	0	0	1	0	0	0	0
21539	20080044	대정전의 밤에	2008-01-29	2	2	일본	2008	0	0	0	0	0	0	1	0

3319 rows × 29 columns

In [94]:

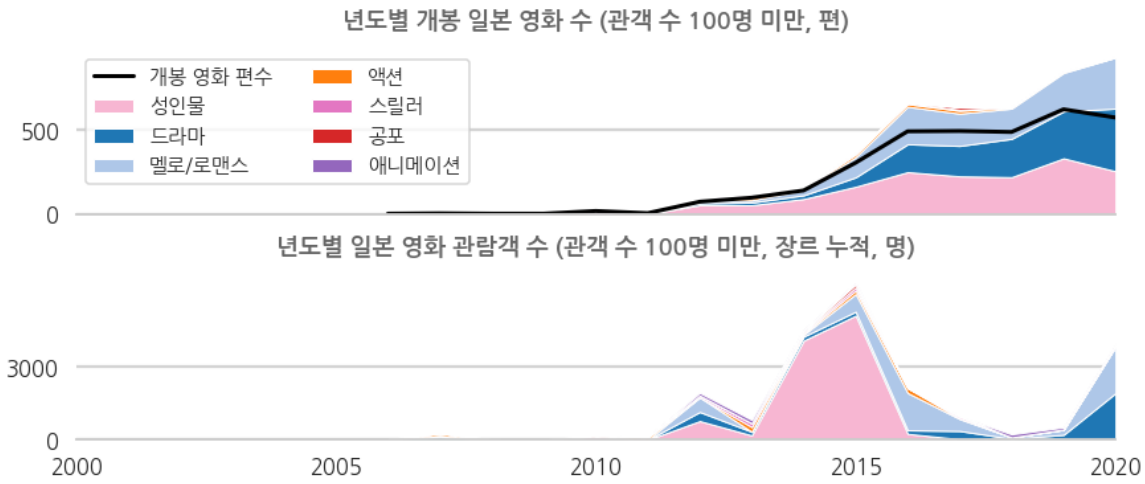
```
df_bo_genresY_jp100 = df_bo_genres_jp.loc[df_bo_genres_jp["서울 관객수"] < 100].groupby("openYear").sum().reset_index()
display(df_bo_genresY_jp100.head(3))
print(df_bo_genresY_jp100.filter(like="G_").sum().sort_values(ascending=False))
```

	openYear	서울 관객수	전국 관객수	G_SF	G_가족	G_공연	G_공포	G_기타	G_다큐멘터리	G_드라마	G_멜로/로맨스	G_뮤지컬	G_미스터리	G_범죄	G_사극	G_서부극	G_성인물
0	2006	117	117	0	0	0	0	0	0	1	0	0	1	0	0	0	0
1	2007	212	212	0	0	0	0	0	0	3	1	0	0	0	0	0	0
2	2008	13	24	0	0	0	1	0	0	1	0	0	0	0	0	0	0

```
G_성인물          1660
G_드라마          1344
G_멜로/로맨스       1307
G_액션             88
G_기타             55
G_스릴러            33
G_공포             30
G_코미디            24
G_애니메이션          24
G_범죄             16
G_SF             12
G_미스터리           10
G_어드벤처            5
G_판타지             5
G_다큐멘터리           3
G_사극             2
G_가족             1
G_서부극             0
G_뮤지컬             0
G_전쟁             0
G_공연             0
dtype: int64
```

In [95]:

```
fig, axs = plt.subplots(figsize=(12, 5), nrows=2, sharex=True, constrained_layout=True)

# 장르별 영화 개봉 편수
axs[0].plot(df_bo_genres_jp.loc[df_bo_genres_jp["서울 관객수"] < 100].groupby("openYear").count
().reset_index()["openYear"],
         df_bo_genres_jp.loc[df_bo_genres_jp["서울 관객수"] < 100].groupby("openYear").count().
reset_index()["서울 관객수"],
         c="k", lw=3, label="개봉 영화 편수", zorder=2)
axs[0].stackplot(df_bo_genresY_jp100["openYear"],
                 [df_bo_genresY_jp100["G_성인물"], df_bo_genresY_jp100["G_드라마"], df_bo_genres
Y_jp100["G_멜로/로맨스"], df_bo_genresY_jp100["G_액션"], df_bo_genresY_jp100["G_스릴러"], df_bo_
genresY_jp100["G_공포"], df_bo_genresY_jp100["G_애니메이션"]],
                 colors=[c_ero, c_drama, c_romance, c_action, c_thriller, c_horror, c_ani], lw=1
,
                 labels=["성인물", "드라마", "멜로/로맨스", "액션", "스릴러", "공포", "애니메이
션"])
axs[0].set_xlim(2000, 2020)
xticks = [2000, 2005, 2010, 2015, 2020]
axs[0].set_xticks(xticks)
axs[0].set_xticklabels(xticks)
axs[0].grid(False)
yticks = [0, 500]
axs[0].set_yticks(yticks)
axs[0].set_yticklabels(yticks)
for y in yticks:
    axs[0].axhline(y, c="lightgray", zorder=-1)
axs[0].spines[["left", "top", "right"]].set_visible(False)
axs[0].set_title("년도별 개봉 일본 영화 수 (관객 수 100명 미만, 편)", pad=16, fontdict=font_titl
e)
axs[0].legend(loc="upper left", fontsize="small", ncol=2)

# 관객 수
axs[1].set_title("년도별 일본 영화 관람객 수 (관객 수 100명 미만, 장르 누적, 명)", pad=16, fontd
ict=font_title)

df_bo_genres_jp100P = deepcopy(df_bo_genres_jp.loc[df_bo_genres_jp["서울 관객수"] < 100])
for g in genres_order:
    df_bo_genres_jp100P[f"G_{g}"] = df_bo_genres_jp100P[f"G_{g}"] * df_bo_genres_jp100P["서울
 관객수"]
df_bo_genresY_jp100P = df_bo_genres_jp100P.groupby("openYear").sum().filter(like="G_").reset_ind
ex()
axs[1].stackplot(df_bo_genresY_jp100P["openYear"],
                 [df_bo_genresY_jp100P["G_성인물"], df_bo_genresY_jp100P["G_드라마"], df_bo_genr
esY_jp100P["G_멜로/로맨스"], df_bo_genresY_jp100P["G_액션"], df_bo_genresY_jp100P["G_스릴러"], d
f_bo_genresY_jp100P["G_공포"], df_bo_genresY_jp100P["G_애니메이션"]],
                 colors=[c_ero, c_drama, c_romance, c_action, c_thriller, c_horror, c_ani], lw=1
,
                 labels=["성인물", "드라마", "멜로/로맨스", "액션", "스릴러", "공포", "애니메이
션"])
axs[1].spines[["left", "top", "right"]].set_visible(False)
axs[1].grid(False)
yticks = [0, 3000]
axs[1].set_yticks(yticks)
axs[1].set_yticklabels(yticks)
for y in yticks:
    axs[1].axhline(y, c="lightgray", zorder=-1)

fig.savefig("./images/bo_genres_jp.png", dpi=200)
```

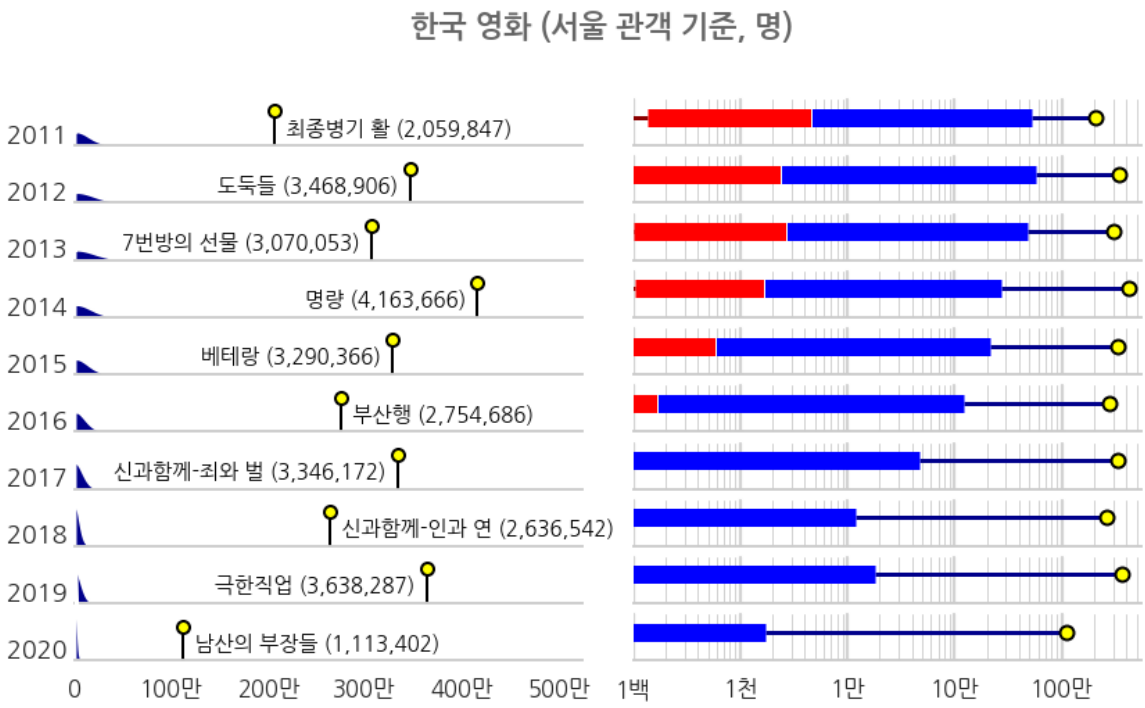


3.4.4. 2011-2020

3.4.4.1. 한국영화

In [96]:

```
plot_boS_year(2011, 2020, '한국', annot_pos=0.6, xmax=5e6)
```



In [97]:

```
# 한국영화 장르 변화 확인 : 총 7352편
df_bo_genres_kr = df_bo_genres.query("국적 == '한국'")
print(df_bo_genres_kr.shape)
df_bo_genres_kr.head()
```

(7352, 29)

Out[97]:

	movieCd	영화명	개봉일	서울 관객수	전국 관객수	국적	openYear	G_SF	G_가족	G_공연	G_공포	G_기타	G_다큐멘터리	G_드라마
1	20201002	조제	2020-12-10	45870	186647	한국	2020	0	0	0	0	0	0	1
6	20050355	음란서생	2006-02-23	868692	2576022	한국	2006	0	0	0	0	0	0	0
11	19860031	변강쇠	1986-05-03	107982	0	한국	1986	0	0	0	0	0	0	1
12	19820012	애마부인	1982-02-06	315738	0	한국	1982	0	0	0	0	0	0	1
22	20101233	평범한 날들	2011-09-29	966	1290	한국	2011	0	0	0	0	0	0	1

In [98]:

```
df_bo_genres_kr.query("G_다큐멘터리 == 1").sort_values("서울 관객수", ascending=False).head(10)
```

Out[98]:

	movieCd	영화명	개봉일	서울 관객수	전국 관객수	국적	openYear	G_SF	G_가족	G_공연	G_공포	G_기타	G_다큐멘터리
6286	20141111	님아, 그 강을 건너지 마오	2014-11-27	1140795	4801355	한국	2014	0	0	0	0	0	1
10202	20081727	워낭소리	2009-01-15	958688	2962897	한국	2009	0	0	0	0	0	1
7415	20174602	노무현입니다	2017-05-25	427607	1854867	한국	2017	0	0	0	0	0	1
1491	20100962	울지마, 톤즈	2010-09-09	174478	441707	한국	2010	0	0	0	0	0	1
5974	20182352	그날, 바다	2018-04-12	136574	540558	한국	2018	0	0	0	0	0	1
6950	20179187	공범자들	2017-08-17	92054	258478	한국	2017	0	0	0	0	0	1
3498	20100027	회복	2010-01-14	66450	157793	한국	2010	0	0	0	0	0	1
7889	20162545	무현, 두 도시 이야기	2016-10-26	63745	193578	한국	2016	0	0	0	0	0	1

	movieCd	영화명	개봉일	서울 관객수	전국 관객수	국적	openYear	G_SF	G_가족	G_공연	G_공포	G_기타	G_다큐멘터리
6258	20173301	서서평, 천천히 평온하게	2017-04-26	57346	123036	한국	2017	0	0	0	0	0	1
9440	20163041	자백	2016-10-13	55388	129950	한국	2016	0	0	0	0	0	1

In [99]:

```
# 서울 관객 100명 이하 영화: 총 1945편
df_bo_genres_kr.loc[df_bo_genres_kr["서울 관객수"] < 100]
```

Out[99]:

	movieCd	영화명	개봉일	서울관객수	전국관객수	국적	openYear	G_SF	G_가족	G_공연	G_공포	G_기타	G_다큐멘터리	G_드라마	G_멜로로맨스
77	20182466	퀴어영화 뷰티풀	2018-10-06	65	65	한국	2018	0	0	0	0	0	0	0	1
82	20184541	2017 동명이인 프로젝트	2018-05-25	0	11	한국	2018	0	0	0	0	1	0	0	0
83	20183137	동명이인 프로젝트 시즌2	2019-07-04	0	130	한국	2019	0	0	0	0	1	0	0	0
121	20196653	유정-스며들다	2019-04-24	3	39	한국	2019	0	0	0	0	0	0	0	1
195	20207344	남편친구	2020-12-03	60	60	한국	2020	0	0	0	0	0	0	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
21390	19980008	파파라치	1999-04-10	98	0	한국	1999	0	0	0	0	0	0	1	0
21769	20081367	가벼운잠	2008-10-23	70	149	한국	2008	0	0	0	0	0	0	1	0
21801	20112632	위도	2011-10-20	69	276	한국	2011	0	0	0	0	0	0	0	0
21915	20110004	돌아오는 길	2011-04-07	65	163	한국	2011	0	0	0	0	0	0	1	0
22826	19870030	잊을수없는순간	1987-06-01	80	0	한국	1987	0	0	0	0	1	0	0	0

1945 rows × 29 columns

In [100]:

```
df_bo_genresY_kr100 = df_bo_genres_kr.loc[df_bo_genres_kr["서울 관객수"] < 100].groupby("openYear").sum().reset_index()
display(df_bo_genresY_kr100.head(3))
print(df_bo_genresY_kr100.filter(like="G_").sum().sort_values(ascending=False))
```

	openYear	서울관객수	전국관객수	G_SF	G_가족	G_공연	G_공포	G_기타	G_다큐멘터리	G_드라마	G_멜로/로맨스	G_뮤지컬	G_미스터리	G_범죄	G_사극	G_서부극	G_성인물	G_스릴러
0	1973	20	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
1	1979	84	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0
2	1987	80	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0

```
G_멜로/로맨스      1465
G_드라마           434
G_성인물           149
G_기타             75
G_코미디            62
G_스릴러            46
G_다큐멘터리          22
G_액션             19
G_공포             15
G_미스터리           14
G_판타지            10
G_애니메이션           5
G_범죄              5
G_가족              4
G_공연              3
G_SF              2
G_뮤지컬             1
G_서부극             0
G_사극              0
G_어드벤처            0
G_전쟁              0
dtype: int64
```

In [101]:

```
df_grade = df_bo_genres_kr.loc[df_bo_genres_kr["서울 관객수"] < 100].query("G_성인물 != 1").loc[
df_bo_genres_kr["G_멜로/로맨스"]==1].query("openYear > 2000")
df_grade["영화명_국적_openYear"] = df_grade[["영화명", "국적", "openYear"]].apply(lambda x: f"
{x[0]}_{x[1]}_{x[2]}", axis=1)
df_grade
```

Out[101]:

	movieCd	영화명	개봉일	서울관객수	전국관객수	국적	openYear	G_SF	G_가족	G_공연	G_공포	G_기타	G_다큐멘터리	G_드라마	G_멜로/로맨스
77	20182466	퀴어영화뷰티풀	2018-10-06	65	65	한국	2018	0	0	0	0	0	0	0	1
121	20196653	유정-스며들다	2019-04-24	3	39	한국	2019	0	0	0	0	0	0	0	1
332	20157434	베드(감독판)	2015-03-12	54	54	한국	2015	0	0	0	0	0	0	1	1
335	20204171	며느리:탐스러운육체	2020-12-29	0	1	한국	2020	0	0	0	0	0	0	1	1
340	20203210	더러운아내무삭제판	2020-12-23	60	60	한국	2020	0	0	0	0	0	0	1	1
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
16269	20122025	금지된섹스불륜2	2012-11-15	0	10	한국	2012	0	0	0	0	0	0	1	1

	movieCd	영화명	개봉일	서울관객수	전국관객수	국적	openYear	G_SF	G_가족	G_공연	G_공포	G_기타	G_다큐멘터리	G_드라마	G_멜로/로맨스
16296	20122707	스무살의이중생활	2012-11-29	0	10	한국	2012	0	0	0	0	0	0	1	1
16308	20123362	몽정애-감독판	2012-12-20	0	20	한국	2012	0	0	0	0	0	0	1	1
16312	20123365	전망좋은해변-두여자	2012-12-26	0	40	한국	2012	0	0	0	0	0	0	1	1
19654	20101282	패밀리마트	2010-11-25	80	165	한국	2010	0	0	0	0	0	0	1	1

1446 rows × 30 columns

In [102]:

```
df_bo_tmp = deepcopy(df_bo)
df_bo_tmp["영화명"] = df_bo_tmp["영화명"].str.replace(" ", "")
# df_bo_tmp["영화명_국적_openYear"] = df_bo[["영화명", "국적", "openYear"]].apply(lambda x: f"{x[0]}_{x[1]}_{x[2]}", axis=1)
df_grade = df_grade.merge(df_bo_tmp[["영화명", "등급"]], on="영화명")
df_grade.head()
```

Out[102]:

	movieCd	영화명	개봉일	서울관객수	전국관객수	국적	openYear	G_SF	G_가족	G_공연	G_공포	G_기타	G_다큐멘터리	G_드라마	G_멜로/로맨스	G_뮤지컬
0	20182466	퀴어영화 뷰티풀	2018-10-06	65	65	한국	2018	0	0	0	0	0	0	0	1	0
1	20196653	유정-스며들다	2019-04-24	3	39	한국	2019	0	0	0	0	0	0	0	1	0
2	20157434	베드(감독판)	2015-03-12	54	54	한국	2015	0	0	0	0	0	0	1	1	0
3	20204171	며느리:탐스러운육체	2020-12-29	0	1	한국	2020	0	0	0	0	0	0	1	1	0
4	20203210	더러운아내무삭제판	2020-12-23	60	60	한국	2020	0	0	0	0	0	0	1	1	0

In [103]:

```
df_grade.drop_duplicates("movieCd", inplace=True)
df_grade.shape
```

Out[103]:

(1446, 31)

In [104]:

```
df_grade.head()
```

Out[104]:

	movieCd	영화명	개봉일	서울관객수	전국관객수	국적	openYear	G_SF	G_가족	G_공연	G_공포	G_기타	G_다큐멘터리	G_드라마	G_멜로/로맨스	G_뮤지컬
0	20182466	퀴어영화뷰티풀	2018-10-06	65	65	한국	2018	0	0	0	0	0	0	0	1	0
1	20196653	유정-스며들다	2019-04-24	3	39	한국	2019	0	0	0	0	0	0	0	1	0
2	20157434	베드(감독판)	2015-03-12	54	54	한국	2015	0	0	0	0	0	0	1	1	0
3	20204171	며느리:탐스러운육체	2020-12-29	0	1	한국	2020	0	0	0	0	0	0	1	1	0
4	20203210	더러운아내무삭제판	2020-12-23	60	60	한국	2020	0	0	0	0	0	0	1	1	0

In [105]:

```
df_grade["등급"] = df_grade["등급"].map({"15세이상관람가":"15",
                                    "12세이상관람가":"12",
                                    "전체관람가":"A",
                                    "12세관람가":"12",
                                    "청소년관람불가":"18",
                                    "18세관람가":"18",
                                    "15세 미만인 자는 관람할 수 없는 등급":"15",
                                    '12세이상관람가,12세관람가':"12",
                                    '15세관람가,15세이상관람가':"15",
                                    '청소년관람불가,15세이상관람가':"15",
                                    '18세관람가,15세이상관람가':"15",
                                    '18세관람가,청소년관람불가':"18",
                                    '12세이상관람가,전체관람가':"A",
                                    '12세이상관람가,15세이상관람가':"12",
                                    '15세이상관람가,전체관람가':"A",
                                    '제한상영가':"X",
                                    '15세관람가,12세이상관람가':"12",
                                    '고등학생이상관람가':"15",
                                    '연소자관람불가':"18",
                                    '모든 관람객이 관람할 수 있는 등급':"A",
                                    '중학생이상관람가':"12",
                                    '연소자관람가':"A",
                                    '국민학생관람불가':"15",
                                    '미성년자관람불가':"18",
                                    '18세 미만인 자는 관람할 수 없는 등급':"18",
                                    '12세이상관람가,연소자관람가':"A",
                                    '연소자관람불가,15세이상관람가':"15",
                                    '연소자관람가,15세이상관람가':"A",
                                    '15세이상관람가,중학생이상관람가':"12",
                                    '12세 미만인 자는 관람할 수 없는 등급':"12",
                                    '15세 미만인 자는 관람할 수 없는 등급 ,15세이상관람가':"15",
                                    '연소자관람불가,청소년관람불가':"18",
                                    '연소자관람가,전체관람가':"A",
                                    '15세이상관람가,18세 미만인 자는 관람할 수 없는 등급':"15",
                                    '고등학생이상관람가,15세이상관람가':"15",
                                    '15세이상관람가,미성년자관람불가':"15",
                                    '미성년자관람가':"A",
                                    '국민학생관람불가,15세이상관람가':"12",
                                    '12세이상관람가,중학생이상관람가':"12",
                                    '고등학생이상관람가,청소년관람불가':"15",
                                    '12세이상관람가,국민학생관람불가,15세이상관람가':"12",
                                    '12세이상관람가,국민학생관람불가':"12",
                                    '12세이상관람가,연소자관람가,전체관람가':"A",
                                    '청소년관람불가,12세관람가':"12",
                                    '국민학생관람불가,중학생이상관람가':"12",
                                    '국민학생관람불가,청소년관람불가':"12",
                                    '청소년관람불가,전체관람가':"A",
                                    '모든 관람객이 관람할 수 있는 등급,전체관람가':"A",
                                    '12세이상관람가,15세 미만인 자는 관람할 수 없는 등급':"12",
                                    '전체관람가,중학생이상관람가':"A",
                                    '청소년관람불가,고등학생이상관람가':"15",
                                    '12세이상관람가,고등학생이상관람가':"12"
                                    })
```

In [106]:

```
print(df_grade["등급"].shape)
df_grade["등급"].value_counts(dropna=False)
```

(1446,)

Out[106]:

```
18    1436
15       6
12       3
A        1
Name: 등급, dtype: int64
```

In [107]:

```
df_grade = df_grade.dropna(subset=["등급"])
for g in ["A", "12", "15", "18"]:
    df_grade[f"등급_{g}"] = [0] * df_grade.shape[0]
    df_grade.loc[df_grade["등급"] == g, f"등급_{g}"] = 1
df_grade.drop("등급", inplace=True, axis=1)
```

In [108]:

```
df_grade.head()
```

Out[108]:

	movieCd	영화명	개봉일	서울관객수	전국관객수	국적	openYear	G_SF	G_가족	G_공연	G_공포	G_기타	G_다큐멘터리	G_드라마	G_멜로/로맨스	G_뮤지컬
0	20182466	퀴어영화 뷰티풀	2018-10-06	65	65	한국	2018	0	0	0	0	0	0	0	1	0
1	20196653	유정-스며들다	2019-04-24	3	39	한국	2019	0	0	0	0	0	0	0	1	0
2	20157434	베드(감독판)	2015-03-12	54	54	한국	2015	0	0	0	0	0	0	1	1	0
3	20204171	며느리: 탐스러운 육체	2020-12-29	0	1	한국	2020	0	0	0	0	0	0	1	1	0
4	20203210	더러운 아내 무삭제판	2020-12-23	60	60	한국	2020	0	0	0	0	0	0	1	1	0

In [109]:

```
df_grade.query("등급_18 == 0")
```

Out[109]:

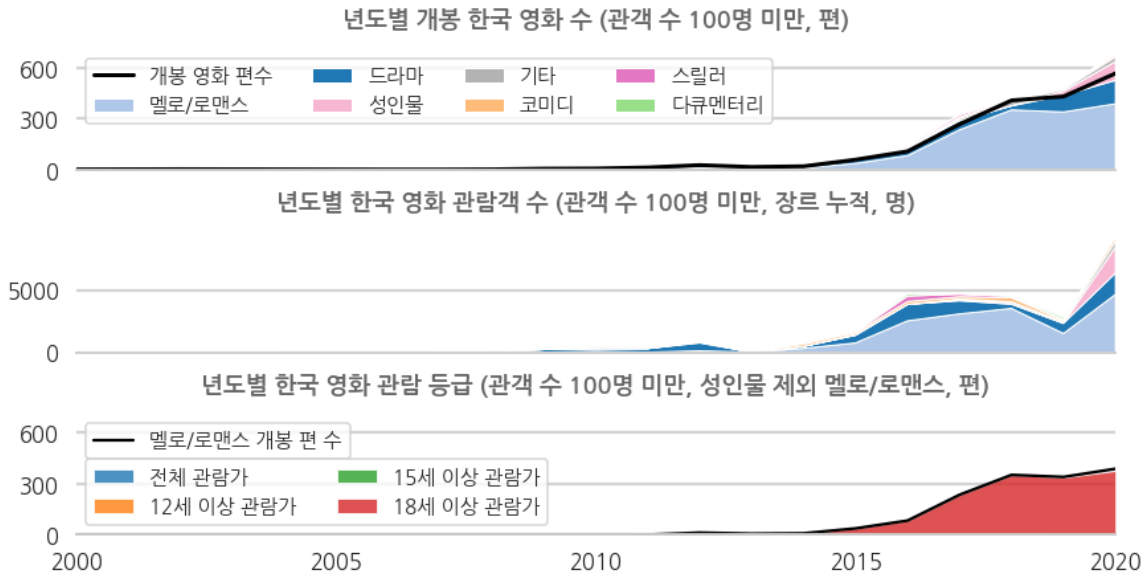
	movieCd	영화명	개봉일	서울관객수	전국관객수	국적	openYear	G_SF	G_가족	G_공연	G_공포	G_기타	G_다큐멘터리	G_드라마	G_멜로/로맨스	G_
837	20186663	뷰티풀뱀파이어	2018-10-08	1	62	한국	2018	0	0	0	0	0	0	1	1	
896	20172625	천사의시간	2018-06-21	14	31	한국	2018	0	0	0	0	0	0	1	1	
926	20183129	애연	2018-06-28	4	39	한국	2018	0	0	0	0	0	0	0	1	
952	20174203	열대야	2018-05-24	0	5	한국	2018	0	0	0	0	0	0	1	1	
1100	20177925	심장박동조작극	2017-12-21	9	76	한국	2017	0	0	0	0	0	0	0	1	
1228	20178665	이사랑도전해질까요	2017-06-21	0	1	한국	2017	0	0	0	0	0	0	1	1	
1350	20158705	서울야행	2016-05-05	21	53	한국	2016	0	0	0	0	0	0	1	1	
1356	20122235	찡찡막막	2016-11-24	0	51	한국	2016	0	0	0	0	0	0	1	1	

	movieCd	영화명	개봉일	서울관객수	전국관객수	국적	openYear	G_SF	G_가족	G_공연	G_공포	G_기타	G_다큐멘터리	G_드라마	G_멜로/로맨스	G_
1403	20162581	연애의 발동: 상해여자, 부산남자	2016-06-02	53	93	한국	2016	0	0	0	0	0	0	0	1	
1422	20158151	황진이	2015-12-10	1	1	한국	2015	0	0	0	0	0	0	1	1	

In [110]:

```
fig, axs = plt.subplots(figsize=(12, 6), nrows=3, sharex=True, constrained_layout=True)

# 장르별 영화 개봉 편수
axs[0].plot(df_bo_genres_kr.loc[df_bo_genres_kr["서울 관객수"] < 100].groupby("openYear").count
().reset_index()["openYear"],
         df_bo_genres_kr.loc[df_bo_genres_kr["서울 관객수"] < 100].groupby("openYear").count().
reset_index()["서울 관객수"],
         c="k", lw=3, label="개봉 영화 편수", zorder=2)
axs[0].stackplot(df_bo_genresY_kr100["openYear"],
                 [df_bo_genresY_kr100["G_멜로/로맨스"], df_bo_genresY_kr100["G_드라마"], df_bo_g
enresY_kr100["G_성인물"], df_bo_genresY_kr100["G_기타"],
                  df_bo_genresY_kr100["G_코미디"], df_bo_genresY_kr100["G_스릴러"], df_bo_genres
Y_kr100["G_다큐멘터리"]],
                 colors=[c_romance, c_drama, c_ero, c_etc, c_comedy, c_thriller, c_docu], lw=1,
                 labels=["멜로/로맨스", "드라마", "성인물", "기타", "코미디", "스릴러", "다큐멘
터리"])
axs[0].set_xlim(2000, 2020)
xticks = [2000, 2005, 2010, 2015, 2020]
axs[0].set_xticks(xticks)
axs[0].set_xticklabels(xticks)
axs[0].grid(False)
yticks = [0, 300, 600]
axs[0].set_yticks(yticks)
axs[0].set_yticklabels(yticks)
for y in yticks:
    axs[0].axhline(y, c="lightgray", zorder=-1)
axs[0].spines[["left", "top", "right"]].set_visible(False)
axs[0].set_title("년도별 개봉 한국 영화 수 (관객 수 100명 미만, 편)", pad=16, fontdict=font_titl
e)
axs[0].legend(loc="upper left", fontsize="small", ncol=4)

# 관객 수
axs[1].set_title("년도별 한국 영화 관람객 수 (관객 수 100명 미만, 장르 누적, 명)", pad=16, fontd
ict=font_title)

df_bo_genres_kr100P = deepcopy(df_bo_genres_kr.loc[df_bo_genres_kr["서울 관객수"] < 100])
for g in genres_order:
    df_bo_genres_kr100P[f"G_{g}"] = df_bo_genres_kr100P[f"G_{g}"] * df_bo_genres_kr100P["서울
 관객수"]
df_bo_genresY_kr100P = df_bo_genres_kr100P.groupby("openYear").sum().filter(like="G_").reset_ind
ex()
axs[1].stackplot(df_bo_genresY_kr100P["openYear"],
                 [df_bo_genresY_kr100P["G_멜로/로맨스"], df_bo_genresY_kr100P["G_드라마"], df_bo
_genresY_kr100P["G_성인물"], df_bo_genresY_kr100P["G_기타"],
                  df_bo_genresY_kr100P["G_코미디"], df_bo_genresY_kr100P["G_스릴러"], df_bo_genr
esY_kr100P["G_다큐멘터리"]],
                 colors=[c_romance, c_drama, c_ero, c_etc, c_comedy, c_thriller, c_docu], lw=1,
                 labels=["멜로/로맨스", "드라마", "성인물", "기타", "코미디", "스릴러", "다큐멘
터리"])
axs[1].spines[["left", "top", "right"]].set_visible(False)
axs[1].grid(False)
yticks = [0, 5000]
axs[1].set_yticks(yticks)
axs[1].set_yticklabels(yticks)
for y in yticks:
    axs[1].axhline(y, c="lightgray", zorder=-1)

# 등급
axs[2].set_title("년도별 한국 영화 관람 등급 (관객 수 100명 미만, 성인물 제외 멜로/로맨스, 편)",
```

```
pad=16, fontdict=font_title)
df_gradeY = df_grade.groupby("openYear").sum().filter(like="등급_").reset_index()
axs[2].plot(df_bo_genresY_kr100["openYear"], df_bo_genresY_kr100["G_멜로/로맨스"], c="k", label=
"멜로/로맨스 개봉 편 수")
axs[2].stackplot(df_gradeY["openYear"],
                 [df_gradeY["등급_A"], df_gradeY["등급_12"], df_gradeY["등급_15"], df_gradeY["등
급_18"]],
                 labels=["전체 관람가", "12세 이상 관람가", "15세 이상 관람가", "18세 이상 관람
가"], lw=1, alpha=0.8)
handles, labels = axs[2].get_legend_handles_labels()
legend_romance = axs[2].legend(handles=handles[:1], labels=labels[:1], loc="upper left", fontsiz
e="small", ncol=2)
axs[2].legend(handles=handles[1:], labels=labels[1:], loc="lower left", fontsize="small", ncol=2
)
axs[2].add_artist(legend_romance)
axs[2].grid(False)
yticks = axs[0].get_yticks()
axs[2].set_yticks(yticks)
for y in yticks:
    axs[2].axhline(y, c="lightgray", zorder=-1)
axs[2].set_ylim(axs[0].get_ylim())
axs[2].spines[["left", "top", "right"]].set_visible(False)

fig.savefig("./images/bo_genres_kr.png", dpi=200)
```



In [111]:

```
df_grade.query("등급_A == 1")
```

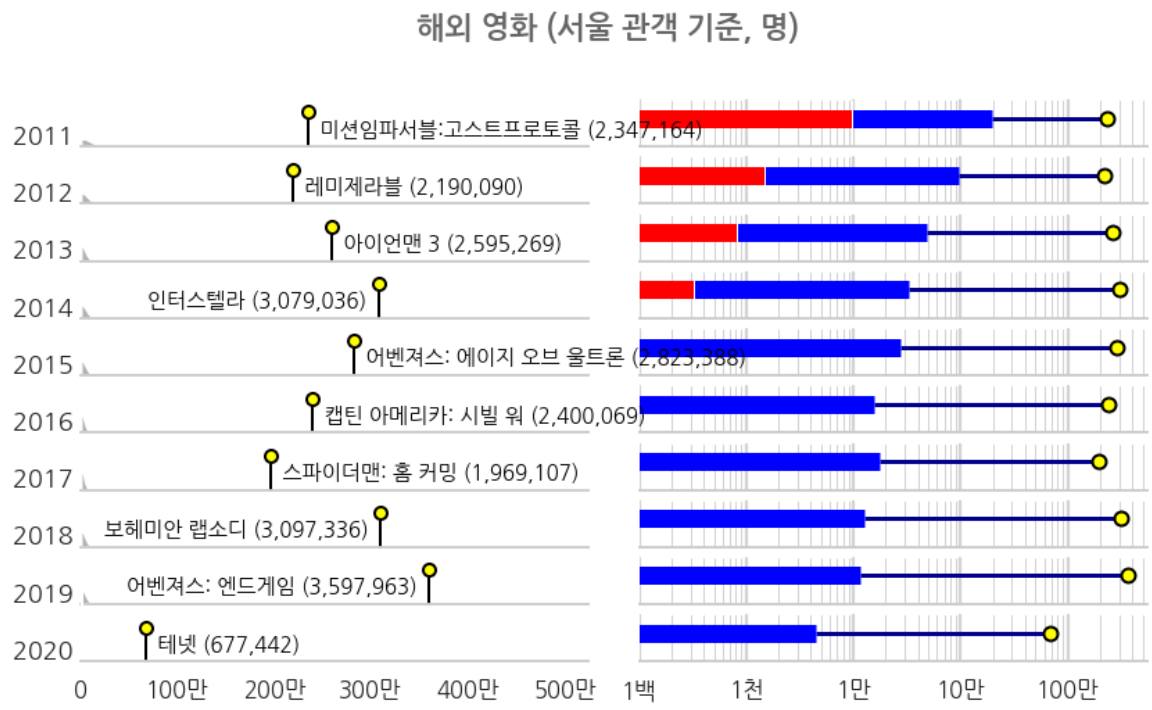
Out[111]:

	movieCd	영화명	개봉일	서울관객수	전국관객수	국적	openYear	G_SF	G_가족	G_공연	G_공포	G_기타	G_다큐멘터리	G_드라마	G_멜로/로맨스	G_뮤지컬
1228	20178665	이사랑도전해질까요	2017-06-21	0	1	한국	2017	0	0	0	0	0	0	1	1	0

3.4.4.2. 해외영화

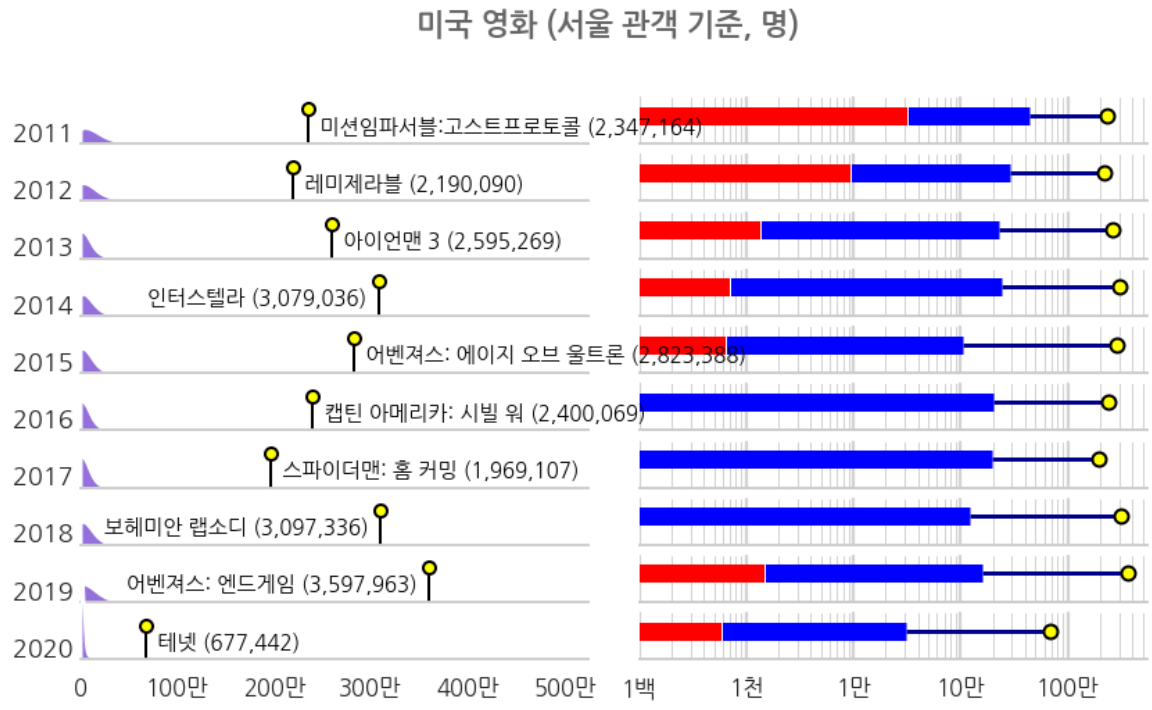
In [112]:

```
plot_boS_year(2011, 2020, '해외', annot_pos=0.6, xmax=5e6)
```



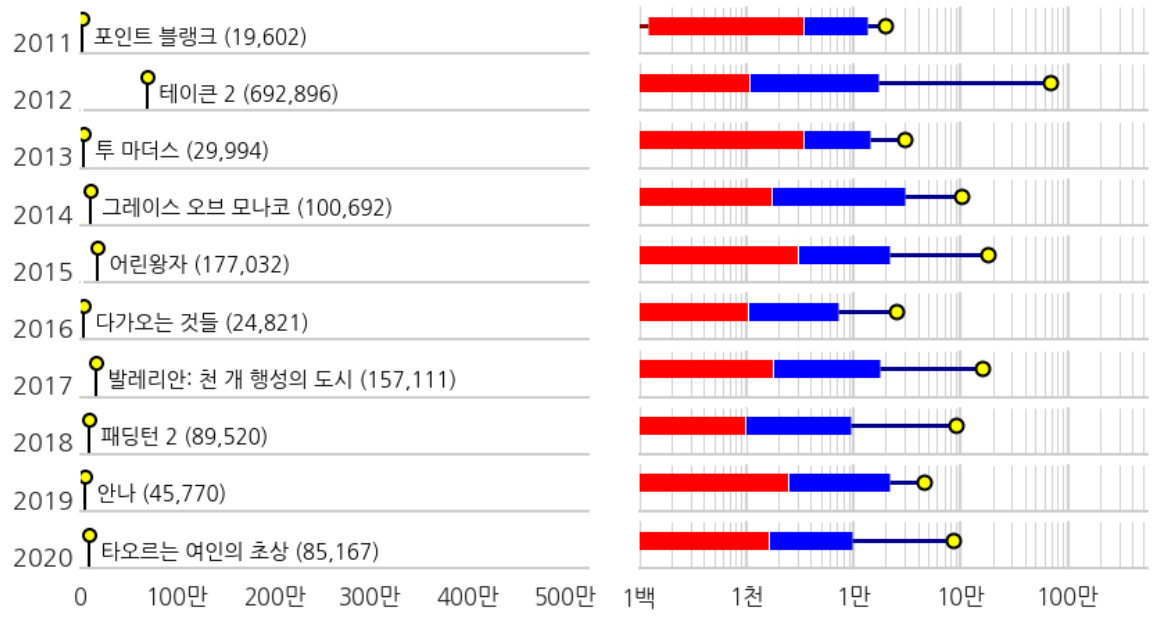
In [113]:

```
plot_boS_year(2011, 2020, '미국', annot_pos=0.6, xmax=5e6)
```



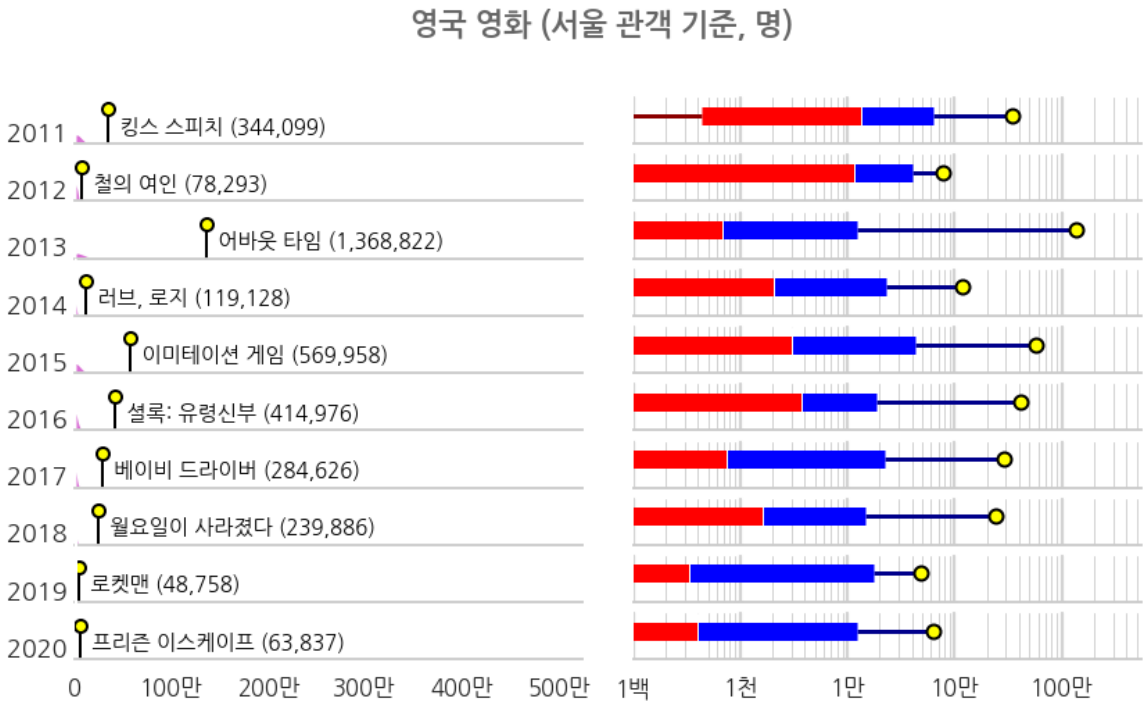
In [114]:

```
plot_boS_year(2011, 2020, '프랑스', annot_pos=0.6, xmax=5e6)
```



In [115]:

```
plot_boS_year(2011, 2020, '영국', annot_pos=0.6, xmax=5e6)
```



3.5. 007영화

In [116]:

```
df_007 = df_bo.loc[df_bo["영화명"].str.contains("007") > 0].sort_values("openYear").loc[~df_bo[
"영화명"].str.contains("2007", na=False, case=False)]
df_007.head()
```

Out[116]:

	순번	영화명	감독	제작사	수입사	배급사	개봉일	영화유형	영화형태	국적	전국스크린수	전국매출액	전국관객수	서울매출액	서울관객수	장르	등급
23231	23232	007다이아몬드는영원히	가이해밀튼	유나이티드아티스트	우성사	NaN	1972-02-29	개봉영화	장편	영국	0	0.0	0	0.0	176386	액션	NaN
23241	23242	007제2편:위기일발	테렌스영	이온프로덕션	(주)삼영필림	NaN	1973-09-21	개봉영화	장편	영국	0	0.0	0	0.0	73330	액션	NaN
23242	23243	007제8편:죽느냐사느냐	가이해밀튼	이온프로덕션	(주)동아수출공사	NaN	1974-01-01	개봉영화	장편	미국	0	0.0	0	0.0	159605	범죄	15세이상관람가
23243	23244	007제9편:황금총을가진사나이	가이해밀튼	유나이티드아티스트	우성사	NaN	1975-01-31	개봉영화	장편	영국	0	0.0	0	0.0	194211	액션	15세이상관람가
23229	23230	007나를사랑한스파이	루이스길버트	다냐크프로덕션,이온프로덕션	한진흥업주식회사	NaN	1978-04-15	개봉영화	장편	영국	0	0.0	0	0.0	545583	스릴러	15세이상관람가

In [117]:

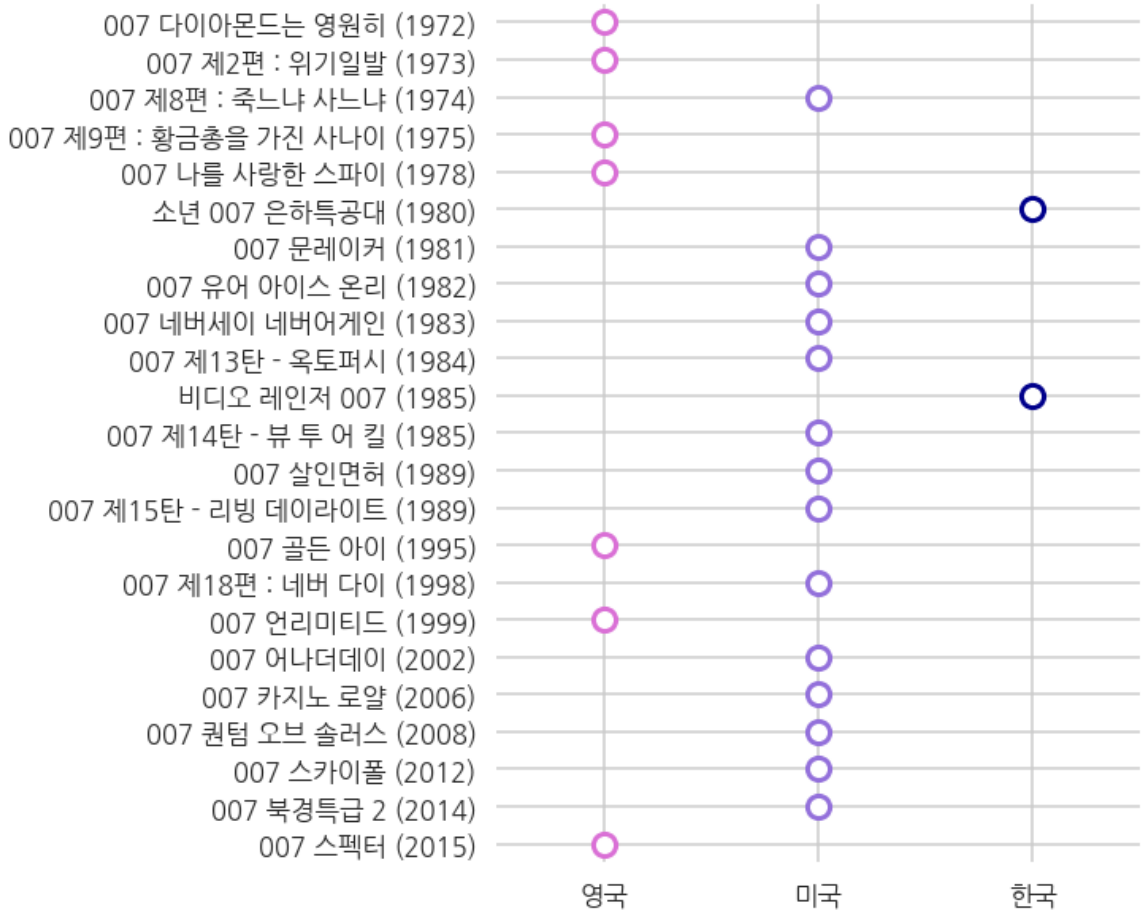
```
df_007.shape
```

Out[117]:

(23, 19)

In [118]:

```
fig, ax = plt.subplots(figsize=(10, 8), constrained_layout=True)

yticks = range(df_007.shape[0])
ax.set_yticks(yticks)
ax.set_yticklabels([f"{x[0]} ({x[2]})" for x in df_007[["영화명", "국적", "openYear"]].values
])
xticklabels = ["영국", "미국", "한국"]
ax.set_xticks([0, 1, 2])
ax.set_xticklabels(xticklabels)
ax.set_xlim(-0.5, 2.5)
ax.set_ylim(-0.5, df_007.shape[0]-0.5)

for i, d in enumerate(df_007[["영화명", "국적", "openYear"]].values):
    if d[1] == "영국":
        x = 0
        c = c_gb
    elif d[1] == "미국":
        x = 1
        c = c_us
    elif d[1] == "한국":
        x = 2
        c = c_kr
    ax.scatter(x, i, lw=3, s=200, fc="w", ec=c)
    ax.spines[["top", "bottom", "left", "right"]].set_visible(False)

ax.invert_yaxis()
fig.savefig("./images/007s.png", dpi=200)
```



3.6. 백만영화의 시대

3.6.1. 전국 관객 수 추정 모델 구축

In [119]:

```
# 머신러닝 모델링 데이터
df_ml = deepcopy(df_bo)
```

In [120]:

```
# 등급 정의
df_ml["등급"].unique()
```

Out[120]:

```
array(['15세이상관람가', '12세이상관람가', '전체관람가', '12세관람가', '15세관람
가', '청소년관람불가',
       '18세관람가', '15세 미만인 자는 관람할 수 없는 등급', '12세이상관람가,12세
관람가',
       '15세관람가,15세이상관람가', nan, '청소년관람불가,15세이상관람가', '18세관
람가,15세이상관람가',
       '18세관람가,청소년관람불가', '12세이상관람가,전체관람가', '12세이상관람가,1
5세이상관람가',
       '15세이상관람가,전체관람가', '제한상영가', '15세관람가,12세이상관람가', '고
등학생이상관람가',
       '연소자관람불가', '모든 관람객이 관람할 수 있는 등급', '중학생이상관람가',
'연소자관람가', '국민학생관람불가',
       '미성년자관람불가', '18세 미만인 자는 관람할 수 없는 등급', '12세이상관람
가,연소자관람가',
       '연소자관람불가,15세이상관람가', '연소자관람가,15세이상관람가', '15세이상관
람가,중학생이상관람가',
       '12세 미만인 자는 관람할 수 없는 등급', '15세 미만인 자는 관람할 수 없는 등
급 ,15세이상관람가', '기타',
       '연소자관람불가,청소년관람불가', '연소자관람가,전체관람가',
       '15세이상관람가,18세 미만인 자는 관람할 수 없는 등급', '미정', '고등학생이
상관람가,15세이상관람가',
       '15세이상관람가,미성년자관람불가', '미성년자관람가', '국민학생관람불가,15세
이상관람가',
       '12세이상관람가,중학생이상관람가', '고등학생이상관람가,청소년관람불가',
       '12세이상관람가,국민학생관람불가,15세이상관람가', '12세이상관람가,국민학생
관람불가',
       '12세이상관람가,연소자관람가,전체관람가', '청소년관람불가,12세관람가', '국
민학생관람불가,중학생이상관람가',
       '국민학생관람불가,청소년관람불가', '청소년관람불가,전체관람가', '모든 관람
객이 관람할 수 있는 등급,전체관람가',
       '12세이상관람가,15세 미만인 자는 관람할 수 없는 등급', '전체관람가,중학생이
상관람가',
       '12세이상관람가,고등학생이상관람가'], dtype=object)
```

In [121]:

```
# 등급 정리
df_ml["등급"] = df_bo["등급"].map({"15세이상관람가":"15",
                                "12세이상관람가":"12",
                                "전체관람가":"A",
                                "12세관람가":"12",
                                "청소년관람불가":"18",
                                "18세관람가":"18",
                                "15세 미만인 자는 관람할 수 없는 등급":"15",
                                '12세이상관람가,12세관람가':"12",
                                '15세관람가,15세이상관람가':"15",
                                '청소년관람불가,15세이상관람가':"15",
                                '18세관람가,15세이상관람가':"15",
                                '18세관람가,청소년관람불가':"18",
                                '12세이상관람가,전체관람가':"A",
                                '12세이상관람가,15세이상관람가':"12",
                                '15세이상관람가,전체관람가':"A",
                                '제한상영가':"X",
                                '15세관람가,12세이상관람가':"12",
                                '고등학생이상관람가':"15",
                                '연소자관람불가':"18",
                                '모든 관람객이 관람할 수 있는 등급':"A",
                                '중학생이상관람가':"12",
                                '연소자관람가':"A",
                                '국민학생관람불가':"15",
                                '미성년자관람불가':"18",
                                '18세 미만인 자는 관람할 수 없는 등급':"18",
                                '12세이상관람가,연소자관람가':"A",
                                '연소자관람불가,15세이상관람가':"15",
                                '연소자관람가,15세이상관람가':"A",
                                '15세이상관람가,중학생이상관람가':"12",
                                '12세 미만인 자는 관람할 수 없는 등급':"12",
                                '15세 미만인 자는 관람할 수 없는 등급 ,15세이상관람가':"15",
                                '연소자관람불가,청소년관람불가':"18",
                                '연소자관람가,전체관람가':"A",
                                '15세이상관람가,18세 미만인 자는 관람할 수 없는 등급':"15",
                                '고등학생이상관람가,15세이상관람가':"15",
                                '15세이상관람가,미성년자관람불가':"15",
                                '미성년자관람가':"A",
                                '국민학생관람불가,15세이상관람가':"12",
                                '12세이상관람가,중학생이상관람가':"12",
                                '고등학생이상관람가,청소년관람불가':"15",
                                '12세이상관람가,국민학생관람불가,15세이상관람가':"12",
                                '12세이상관람가,국민학생관람불가':"12",
                                '12세이상관람가,연소자관람가,전체관람가':"A",
                                '청소년관람불가,12세관람가':"12",
                                '국민학생관람불가,중학생이상관람가':"12",
                                '국민학생관람불가,청소년관람불가':"12",
                                '청소년관람불가,전체관람가':"A",
                                '모든 관람객이 관람할 수 있는 등급,전체관람가':"A",
                                '12세이상관람가,15세 미만인 자는 관람할 수 없는 등급':"12",
                                '전체관람가,중학생이상관람가':"A",
                                '청소년관람불가,고등학생이상관람가':"15",
                                '12세이상관람가,고등학생이상관람가':"12"
                                })
```

In [122]:

```
# 데이터 선별
df_ml = df_ml[["국적", "전국 스크린수", "전국 관객수", "서울 관객수", "장르", "등급", "openYear"
]]

# 전국 관객수 10만 이상, 500만 이하
df_ml = df_ml.loc[df_ml["전국 관객수"] >= 100000].loc[df_ml["전국 관객수"] <= 5000000]

# 결측치 제거
df_ml = df_ml.dropna()

# 데이터 수
df_ml.shape
```

Out[122]:

```
(2091, 7)
```

In [257]:

```
# y : 전국 관객수
y = df_ml["전국 관객수"]

# X: 나머지
X = df_ml.drop("전국 관객수", axis=1)

print(X.shape)
```

```
(2091, 6)
```

In [258]:

```
# LightGBM Pipeline

from sklearn.pipeline import Pipeline
from sklearn.compose import ColumnTransformer
from sklearn.preprocessing import OneHotEncoder
from lightgbm import LGBMRegressor


def model():
    # categorical and numerical features
    cat_features = ["국적", "장르", "등급"]
    cat_transformer = OneHotEncoder(sparse=False, handle_unknown="ignore")

    num_features = ["전국 스크린수", "서울 관객수", "openYear"]
    num_transformer = 'passthrough'

    # 1. 인자 종류별 전처리 적용
    preprocessor = ColumnTransformer(transformers=[("num", num_transformer, num_features),
                                                   ("cat", cat_transformer, cat_features)])

    # 2. 전처리 후 XGBoost 적용
    model = Pipeline(steps=[("preprocessor", preprocessor),
                            ("ml", LGBMRegressor(use_missing=False, random_state=2021))
                           ])

    return model
```

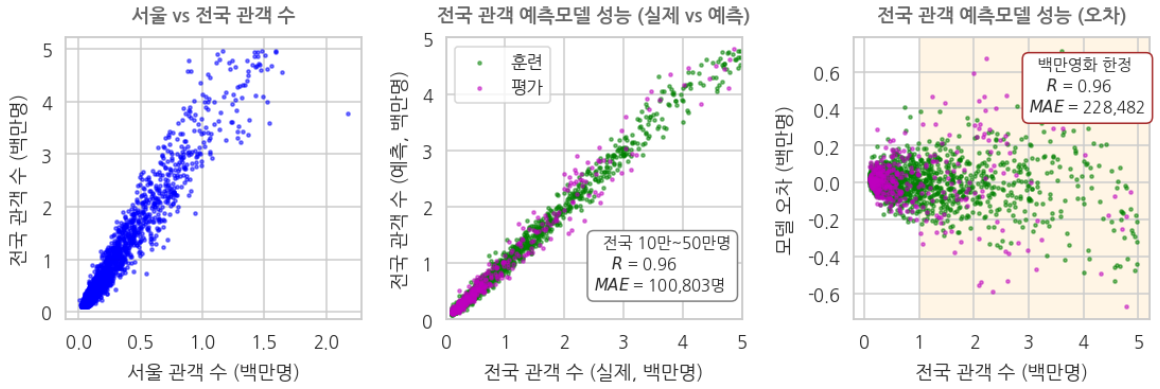
In [259]:

```
# 데이터 나누기
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=2021)

lgbm = model()
lgbm.fit(X_train, y_train)
y_train_pred = lgbm.predict(X_train)
y_test_pred = lgbm.predict(X_test)
```

In [264]:

```
from sklearn.metrics import r2_score
from sklearn.metrics import mean_absolute_error
from scipy import stats

# 전체데이터 예측성능
r2 = r2_score(y_test, y_test_pred)
mae = mean_absolute_error(y_test, y_test_pred)
R = stats.pearsonr(y_test, y_test_pred)

# 백만영화 예측성능
r2_m = r2_score(y_test[y_test>1e6], y_test_pred[y_test>1e6])
mae_m = mean_absolute_error(y_test[y_test>1e6], y_test_pred[y_test>1e6])
R = stats.pearsonr(y_test[y_test>1e6], y_test_pred[y_test>1e6])

fig, axs = plt.subplots(ncols=3, figsize=(15, 5), constrained_layout=True)
axs[0].scatter(df_ml["서울 관객수"]/1e6, y/1e6, s=10, alpha=0.5, c="b")
axs[1].scatter(y_train/1e6, y_train_pred/1e6, s=10, alpha=0.5, c="g", label="훈련")
axs[1].scatter(y_test/1e6, y_test_pred/1e6, s=10, alpha=0.5, c="m", label="평가")
axs[1].legend()
axs[2].scatter(y_train/1e6, (y_train_pred - y_train)/1e6, s=10, alpha=0.5, c="g")
axs[2].scatter(y_test/1e6, (y_test_pred - y_test)/1e6, s=10, alpha=0.5, c="m")

font_title = {"fontweight":"bold", "color":"0.4"}

axs[0].set_yticks([0, 1, 2, 3, 4, 5])
axs[0].set_xlabel("서울 관객 수 (백만명)", labelpad=12)
axs[0].set_ylabel("전국 관객 수 (백만명)", labelpad=12)
axs[0].set_title("서울 vs 전국 관객 수", fontdict=font_title, pad=16)

axs[1].set_xlim(0, 5)
axs[1].set_ylim(0, 5)
axs[1].set_xlabel("전국 관객 수 (실제, 백만명)", labelpad=12)
axs[1].set_ylabel("₩n전국 관객 수 (예측, 백만명)", labelpad=12)
axs[1].set_yticks([0, 1, 2, 3, 4, 5])
axs[1].set_title("전국 관객 예측모델 성능 (실제 vs 예측)", fontdict=font_title, pad=16)

axs[2].axvspan(1, 5.2, fc="orange", zorder=-1, alpha=0.1)
axs[2].set_xlim(-0.2, 5.2)
axs[2].set_xticks([0, 1, 2, 3, 4, 5])
axs[2].set_ylabel("₩n모델 오차 (백만명)", labelpad=12)
axs[2].set_xlabel("전국 관객 수 (백만명)", labelpad=12)
axs[2].set_title("전국 관객 예측모델 성능 (오차)", fontdict=font_title, pad=16)
axs[1].text(2.5, 1.5, "  전국 10만~50만명₩n    $R$ = " + f"{R[0]:.2f}₩n"+"$MAE$ = " + f"{forma
t(int(mae),',')}명", fontsize="small", va="top",
            bbox={"boxstyle":"round", "pad":0.4, "facecolor":"w", "edgecolor":"gray"})
axs[2].text(3, 0.68, "  백만영화 한정₩n    $R$ = " + f"{R[0]:.2f}₩n"+"$MAE$ = " + f"{format(in
t(mae_m), ',')}", fontsize="small", va="top",
            bbox={"boxstyle":"round", "pad":0.4, "facecolor":"w", "edgecolor":"brown"})

fig.savefig("./images/ml.png", dpi=200)

print(mae, mae_m)
```

100803.89538081748 228482.84562281627



In [262]:

```
R
```

Out[262]:

(0.9568587150985421, 1.7290739143782906e-62)

3.6.2. 전국 관객 수 업데이트

In [127]:

```
df_pred = deepcopy(df_bo)
df_pred["등급"] = df_bo["등급"].map({"15세이상관람가":"15",
                                "12세이상관람가":"12",
                                "전체관람가":"A",
                                "12세관람가":"12",
                                "청소년관람불가":"18",
                                "18세관람가":"18",
                                "15세 미만인 자는 관람할 수 없는 등급":"15",
                                '12세이상관람가,12세관람가':"12",
                                '15세관람가,15세이상관람가':"15",
                                '청소년관람불가,15세이상관람가':"15",
                                '18세관람가,15세이상관람가':"15",
                                '18세관람가,청소년관람불가':"18",
                                '12세이상관람가,전체관람가':"A",
                                '12세이상관람가,15세이상관람가':"12",
                                '15세이상관람가,전체관람가':"A",
                                '제한상영가':"X",
                                '15세관람가,12세이상관람가':"12",
                                '고등학생이상관람가':"15",
                                '연소자관람불가':"18",
                                '모든 관람객이 관람할 수 있는 등급':"A",
                                '중학생이상관람가':"12",
                                '연소자관람가':"A",
                                '국민학생관람불가':"15",
                                '미성년자관람불가':"18",
                                '18세 미만인 자는 관람할 수 없는 등급':"18",
                                '12세이상관람가,연소자관람가':"A",
                                '연소자관람불가,15세이상관람가':"15",
                                '연소자관람가,15세이상관람가':"A",
                                '15세이상관람가,중학생이상관람가':"12",
                                '12세 미만인 자는 관람할 수 없는 등급':"12",
                                '15세 미만인 자는 관람할 수 없는 등급 ,15세이상관람가':"15",
                                '연소자관람불가,청소년관람불가':"18",
                                '연소자관람가,전체관람가':"A",
                                '15세이상관람가,18세 미만인 자는 관람할 수 없는 등급':"15",
                                '고등학생이상관람가,15세이상관람가':"15",
                                '15세이상관람가,미성년자관람불가':"15",
                                '미성년자관람가':"A",
                                '국민학생관람불가,15세이상관람가':"12",
                                '12세이상관람가,중학생이상관람가':"12",
                                '고등학생이상관람가,청소년관람불가':"15",
                                '12세이상관람가,국민학생관람불가,15세이상관람가':"12",
                                '12세이상관람가,국민학생관람불가':"12",
                                '12세이상관람가,연소자관람가,전체관람가':"A",
                                '청소년관람불가,12세관람가':"12",
                                '국민학생관람불가,중학생이상관람가':"12",
                                '국민학생관람불가,청소년관람불가':"12",
                                '청소년관람불가,전체관람가':"A",
                                '모든 관람객이 관람할 수 있는 등급,전체관람가':"A",
                                '12세이상관람가,15세 미만인 자는 관람할 수 없는 등급':"12",
                                '전체관람가,중학생이상관람가':"A",
                                '청소년관람불가,고등학생이상관람가':"15",
                                '12세이상관람가,고등학생이상관람가':"12"
                                  })
df_pred["등급"].fillna("18", inplace=True) # 등급 결측치 : 18금
df_pred["장르"].fillna("기타", inplace=True) # 장르 결측치: 기타
df_pred = df_pred.query("openYear < 2010").loc[df_pred["서울 관객수"] >= 200000].loc[df_pred["전
국 관객수"] < 100]
df_pred["전국 관객수"] = lgbm.predict(df_pred[["국적", "전국 스크린수", "서울 관객수", "장르",
```

```
"등급", "openYear"]])
print(df_pred.shape)
```

```
(613, 19)
```

In [128]:

```
# 예측결과 확인
df_pred.loc[df_pred["전국 관객수"] >= 1000000].sort_values("전국 관객수", ascending=False).head
(20)[["영화명", "감독", "개봉일", "국적", "서울 관객수", "전국 관객수"]].rename(columns={"전국
 관객수":"전국 관객수 (추정)"})
```

Out[128]:

	영화명	감독	개봉일	국적	서울 관객수	전국 관객수 (추정)
20482	엽기적인 그녀	곽재용	2001-07-26	한국	1735692	4.412533e+06
21651	집으로...	이정향	2002-04-04	한국	1576943	4.200297e+06
22863	해리포터와 마법사의 돌	크리스 콜럼버스	2001-12-13	미국	1585389	4.184802e+06
18276	반지의 제왕 : 반지원정대	피터 잭슨	2001-12-31	미국	1380734	4.157144e+06
16027	가문의 영광	정흥순	2002-09-12	한국	1605775	4.148018e+06
19630	신라의 달밤	김상진	2001-06-23	한국	1608211	4.148018e+06
21470	조폭마누라	조진규	2001-09-27	한국	1419972	4.148018e+06
18871	사랑과 영혼	제리 주커	1990-11-24	미국	1532589	4.105809e+06
22206	타이타닉	제임스 카메론	1998-02-20	미국	1971780	4.105809e+06
16975	달마야 놀자	박철관	2001-11-08	한국	1253075	4.029885e+06
17777	마이너리티 리포트	스티븐 스필버그	2002-07-25	미국	1401000	3.967194e+06
16251	공동경비구역 JSA	박찬욱	2000-09-08	한국	2513540	3.927797e+06
21923	친구	곽경택	2001-03-31	한국	2678846	3.927797e+06
17283	두사부일체	윤제균,이경원	2001-12-08	한국	1228142	3.722290e+06
19540	스파이더맨	샘 레이미	2002-05-02	미국	1125568	3.705813e+06
18178	미이라	스티븐 소머즈	1999-07-10	미국	1114916	3.705813e+06
22146	클리프 행어	레니 할린	1993-06-12	미국	1118583	3.637584e+06
19845	아마겟돈	마이클 베이	1998-07-03	미국	1170252	3.637584e+06
16250	공공의 적	강우석	2002-01-24	한국	1161500	3.579170e+06
16456	글래디에이터	리들리 스콧	2000-06-03	미국	1242055	3.405340e+06

3.6.3. 장르 결합

In [129]:

```
# 추정치 포함 데이터베이스
df_bo_pred = deepcopy(df_bo)
df_bo_pred.loc[df_pred.index, "전국 관객수"] = df_pred["전국 관객수"]

# 영화명 띄어쓰기 제거
df_bo_pred["영화명"] = df_bo_pred["영화명"].str.replace(' ', '')

# 장르 결합 key
df_bo_pred["영화명_개봉일_대표국적"] = df_bo_pred[["영화명", "개봉일", "국적"]].apply(lambda x:
f"{x[0]}_{x[1]}_{x[2]}", axis=1)
df_bo_pred.head()
```

Out[129]:

	순번	영화명	감독	제작사	수입사	배급사	개봉일	영화유형	영화형태	국적	전국 스크린수	전국 매출액	전국 관객수	서울 매
0	1	명량	김한민	(주)빅스톤픽쳐스	NaN	(주)씨제이이엔엠	2014-07-30	개봉영화	장편	한국	1587	1.357484e+11	17613682.0	3.312123
1	2	극한직업	이병헌	(주)어바웃필름, 영화사해그림주식회사, (주)씨제이이엔엠	NaN	(주)씨제이이엔엠	2019-01-23	개봉영화	장편	한국	1978	1.396480e+11	16264944.0	3.185866
2	3	신과함께-죄와벌	김용화	리얼라이즈픽쳐스(주), (주)덱스터스튜디오	NaN	롯데쇼핑㈜롯데엔터테인먼트	2017-12-20	개봉영화	장편	한국	1912	1.156987e+11	14410754.0	2.753083
3	4	국제시장	윤제균	(주)제이케이필름, (주)씨제이이엔엠	NaN	(주)씨제이이엔엠	2014-12-17	개봉영화	장편	한국	966	1.108280e+11	14245998.0	2.584252

	순번	영화명	감독	제작사	수입사	배급사	개봉일	영화유형	영화형태	국적	전국 스크린수	전국 매출액	전국 관객수	서울 매
4	5	어벤져스: 엔드게임	안소니 루소, 조 루소	NaN	월트디즈니컴퍼니코리아유한책임회사	월트디즈니컴퍼니코리아유한책임회사	2019-04-24	개봉영화	장편	미국	2835	1.221827e+11	13934592.0	3.357714

In [130]:

```
# 장르 결합
df_bo_pred = df_movie_bo.merge(df_bo_pred[["영화명_개봉일_대표국적", "영화명", "개봉일", "전국
 관객수", "국적", "openYear"]])
df_bo_pred = df_bo_pred.merge(df_genres.drop(["movieNm", "openYear"], axis=1), on="movieCd")
df_bo_pred = df_bo_pred.drop(["movieNm", "openDt", "repNationNm", "영화명_개봉일_대표국적"], axi
s=1)
print(df_bo_pred.shape)
df_bo_pred.head()
```

(23182, 28)

Out[130]:

	movieCd	영화명	개봉일	전국 관객수	국적	openYear	G_SF	G_가족	G_공연	G_공포	G_기타	G_다큐멘터리	G_드라마	G_멜로/로맨스	G_뮤지컬
0	20139221	그래비티	2013-10-17	3227452.0	미국	2013	1	0	0	0	0	0	1	0	0
1	20201002	조제	2020-12-10	186647.0	한국	2020	0	0	0	0	0	0	1	1	0
2	20010264	아멜리에	2001-10-19	0.0	프랑스	2001	0	0	0	0	0	0	0	0	0
3	20030039	링 0-버스데이	2003-04-11	5806.0	일본	2003	0	0	0	1	0	0	0	0	0
4	20050265	부에나비스타 소셜 클럽	2001-03-01	0.0	독일	2001	0	0	0	0	0	1	0	0	0

3.6.4. 백만영화, 천만영화

In [131]:

```
# 한국영화
df_bo_kr = df_bo_pred.query("국적 == '한국'")
df_bo_kr_m = df_bo_kr.loc[df_bo_kr["전국 관객수"] >= 1e6] # 백만영화
df_bo_kr_10m = df_bo_kr.loc[df_bo_kr["전국 관객수"] >= 1e7] # 천만영화
```

In [132]:

```
# 한국 백만영화
df_bo_kr_mY_0 = df_bo_kr_m.groupby("openYear")["movieCd"].count()
df_bo_kr_mY_G = df_bo_kr_m.groupby("openYear").sum().filter(like="G_")
df_bo_kr_mY = pd.concat([df_bo_kr_mY_0, df_bo_kr_mY_G], axis=1)
df_bo_kr_mY = df_bo_kr_mY.rename(columns={"movieCd":"영화편수"}).reset_index()
df_bo_kr_mY.tail()
```

Out[132]:

	openYear	영화편수	G_SF	G_가족	G_공연	G_공포	G_기타	G_다큐멘터리	G_드라마	G_멜로/로맨스	G_뮤지컬	G_미스터리	G_범죄	G_사극	G_서부극	G_성인물	G_스릴러	G
34	2016	24	0	0	0	0	0	0	14	0	0	3	4	2	0	0	6	
35	2017	27	0	0	0	0	0	1	10	0	0	2	8	1	0	0	5	
36	2018	29	0	0	0	1	0	0	13	2	0	1	7	4	0	0	4	
37	2019	31	0	0	0	2	0	0	16	2	0	2	7	2	0	0	2	
38	2020	13	0	0	0	0	0	0	7	0	0	1	2	0	0	0	0	

In [133]:

```
# 한국 천만영화
df_bo_kr_10mY_0 = df_bo_kr_10m.groupby("openYear")["movieCd"].count()
df_bo_kr_10mY_G = df_bo_kr_10m.groupby("openYear").sum().filter(like="G_")
df_bo_kr_10mY = pd.concat([df_bo_kr_10mY_0, df_bo_kr_10mY_G], axis=1)
df_bo_kr_10mY = df_bo_kr_10mY.rename(columns={"movieCd":"영화편수"}).reset_index()
df_bo_kr_10mY.tail()
```

Out[133]:

	openYear	영화편수	G_SF	G_가족	G_공연	G_공포	G_기타	G_다큐멘터리	G_드라마	G_멜로/로맨스	G_뮤지컬	G_미스터리	G_범죄	G_사극	G_서부극	G_성인물	G_스릴러	G
8	2015	2	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	
9	2016	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	
10	2017	2	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	
11	2018	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	
12	2019	2	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	

In [134]:

```
# 해외영화
df_bo_nkr = df_bo_pred.query("국적 != '한국'")
df_bo_nkr_m = df_bo_nkr.loc[df_bo_nkr["전국 관객수"] >= 1e6] # 백만영화
df_bo_nkr_10m = df_bo_nkr.loc[df_bo_nkr["전국 관객수"] >= 1e7] # 천만영화
```

In [135]:

```
# 해외 백만영화
df_bo_nkr_mY_0 = df_bo_nkr_m.groupby("openYear")["movieCd"].count()
df_bo_nkr_mY_G = df_bo_nkr_m.groupby("openYear").sum().filter(like="G_")
df_bo_nkr_mY_N = df_bo_nkr_m.groupby("openYear").sum().filter(like="N_")
df_bo_nkr_mY = pd.concat([df_bo_nkr_mY_0, df_bo_nkr_mY_G, df_bo_nkr_mY_N], axis=1)
df_bo_nkr_mY = df_bo_nkr_mY.rename(columns={"movieCd":"영화편수"}).reset_index()
df_bo_nkr_mY.tail()
```

Out[135]:

	openYear	영화편수	G_SF	G_가족	G_공연	G_공포	G_기타	G_다큐멘터리	G_드라마	G_멜로/로맨스	G_뮤지컬	G_미스터리	G_범죄	G_사극	G_서부극	G_성인물	G_스릴러	G
41	2016	27	6	2	0	2	0	0	5	1	2	1	2	0	0	0	2	
42	2017	27	10	1	0	3	0	0	5	2	2	2	1	0	0	0	7	
43	2018	24	10	2	0	1	0	0	3	0	1	0	1	0	0	0	4	
44	2019	19	5	4	0	1	0	0	3	1	0	0	1	0	0	0	4	
45	2020	2	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

In [136]:

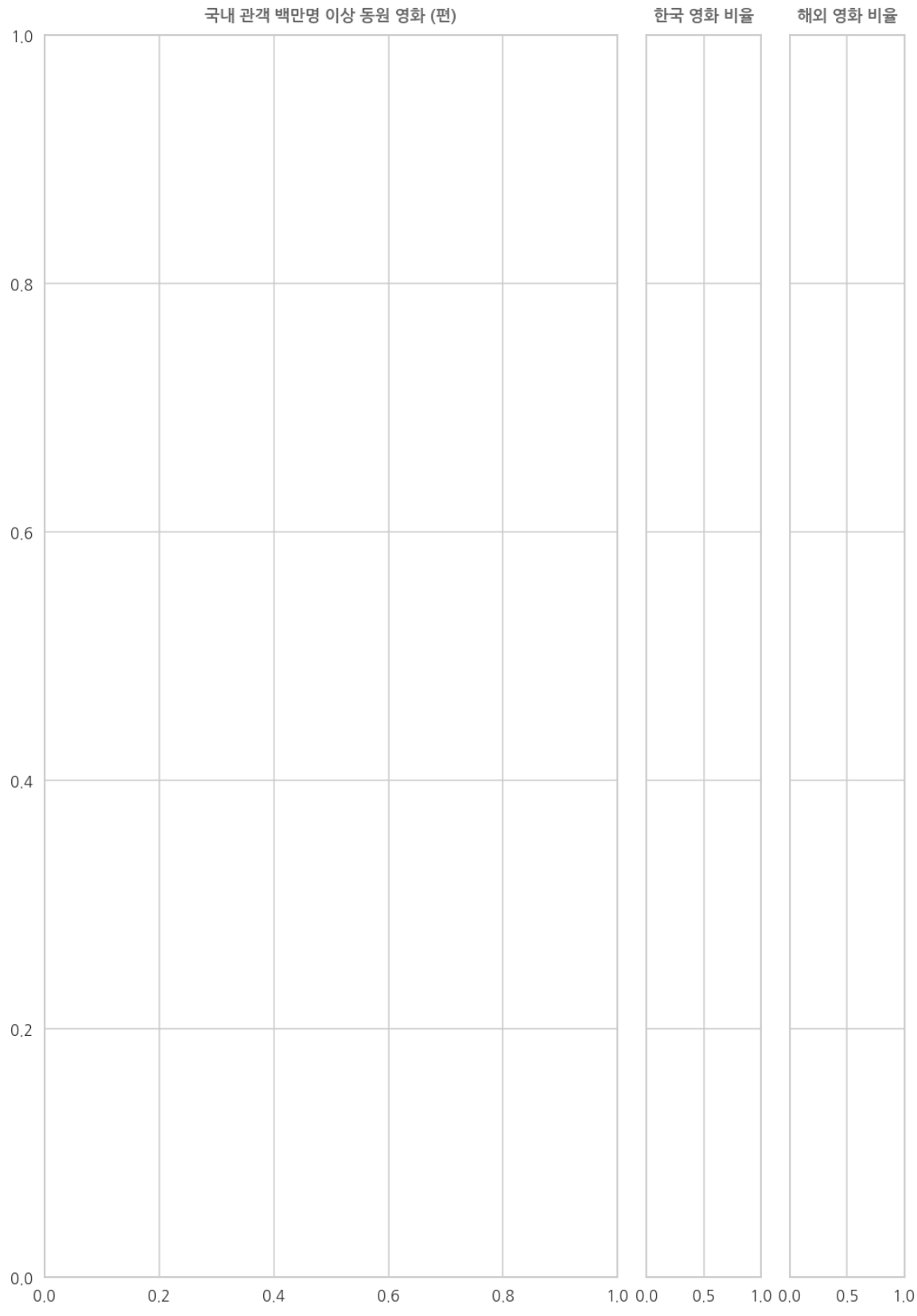
```
# 해외 천만영화
df_bo_nkr_10mY_O = df_bo_nkr_10m.groupby("openYear")["movieCd"].count()
df_bo_nkr_10mY_G = df_bo_nkr_10m.groupby("openYear").sum().filter(like="G_")
df_bo_nkr_10mY_N = df_bo_nkr_10m.groupby("openYear").sum().filter(like="N_")
df_bo_nkr_10mY = pd.concat([df_bo_nkr_10mY_O, df_bo_nkr_10mY_G, df_bo_nkr_10mY_N], axis=1)
df_bo_nkr_10mY = df_bo_nkr_10mY.rename(columns={"movieCd":"영화편수"}).reset_index()
df_bo_nkr_10mY.tail()
```

Out[136]:

	openYear	영화편수	G_SF	G_가족	G_공연	G_공포	G_기타	G_다큐멘터리	G_드라마	G_멜로/로맨스	G_뮤지컬	G_미스터리	G_범죄	G_사극	G_서부극	G_성인물	G_스릴러	G_애니메이션
0	2009	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	2014	2	1	1	0	0	0	0	0	0	1	0	0	0	0	0	0	1
2	2015	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	2018	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	2019	3	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1

In [137]:

```
# 데이터 시각화
fig, axs = plt.subplots(ncols=3, gridspec_kw={"width_ratios":[5,1,1]},
                        sharey=True,
                        figsize=(14, 20), constrained_layout=True)

axs[0].set_title("국내 관객 백만명 이상 동원 영화 (편)", fontdict=font_title, pad=16)
axs[1].set_title("한국 영화 비율", fontdict=font_title, pad=16)
axs[2].set_title("해외 영화 비율", fontdict=font_title, pad=16)

portion_aspect0 = axs[0].get_position().height/axs[0].get_position().width
portion_aspect1 = axs[1].get_position().height/axs[1].get_position().width
portion_aspect2 = axs[2].get_position().height/axs[2].get_position().width
```


국내 관객 백만명 이상 동원 영화 (편)
한국 영화 비율
해외 영화 비율
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0
0.0
0.5
1.0
0.0
0.5
1.0

In [138]:

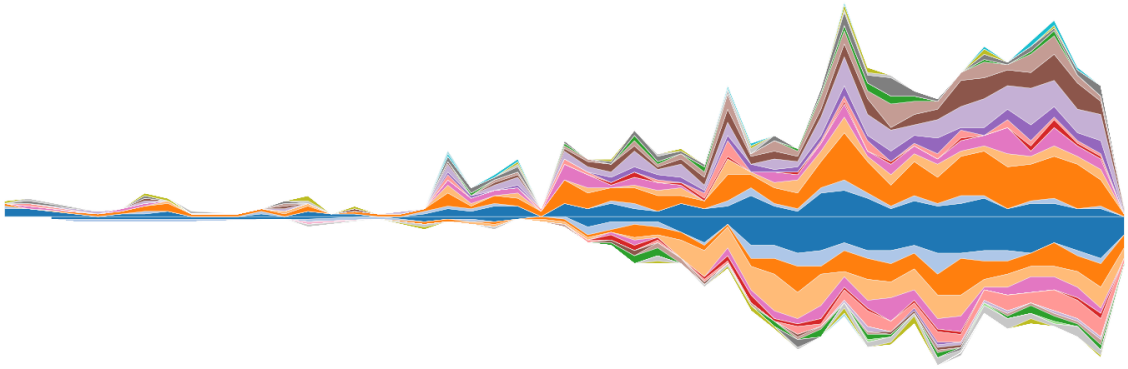
```
genres_noetc = genres_order[:20]
genres_noetc
```

Out[138]:

```
['드라마',
 '멜로/로맨스',
 '액션',
 '코미디',
 '스릴러',
 '성인물',
 '공포',
 '범죄',
 '애니메이션',
 '어드벤처',
 'SF',
 '판타지',
 '미스터리',
 '다큐멘터리',
 '가족',
 '사극',
 '전쟁',
 '공연',
 '뮤지컬',
 '서부극']
```

In [139]:

```
sns.set_palette("tab20")

fig_p0, ax_p0 = plt.subplots(figsize=(axs[0].get_position().height * 20, axs[0].get_position().w
idth * 14), constrained_layout=True)

stack_ys_kr = []
stack_ys_nkr = []
for g in genres_noetc:
    stack_ys_kr.append(eval(f"-df_bo_kr_mY['G_{g}']"))
    stack_ys_nkr.append(eval(f"df_bo_nkr_mY['G_{g}']"))

ax_p0.stackplot(df_bo_kr_mY["openYear"], *stack_ys_kr,
                ec="w", lw=0.4, colors=c_genres[:20])
ax_p0.stackplot(df_bo_nkr_mY["openYear"], *stack_ys_nkr,
                ec="w", lw=0.4, colors=c_genres[:20])

# G_기타
ax_p0.stackplot(df_bo_kr_mY["openYear"],
                -df_bo_kr_mY.filter(like="G_").sum(axis=1),
                ec="w", lw=0.4, zorder=-1)
ax_p0.stackplot(df_bo_nkr_mY["openYear"],
                df_bo_nkr_mY.filter(like="G_").sum(axis=1),
                ec="w", lw=0.4, zorder=-1)
ax_p0.get_children()[40].set_facecolor(c_etc)
ax_p0.get_children()[41].set_facecolor(c_etc)

ax_p0.set_xlim(1971, 2020)
ax_p0.set_ylim(-100, 100)
ax_p0.axis(False)
fig_p0.savefig("./images/bo_year0.png")
```



In [140]:

```
# 국내 genre portion

stack_ys_kr.append(-df_bo_kr_mY['G_기타'])

stack_ys_kr_p = np.array(stack_ys_kr)/np.array(stack_ys_kr).sum(axis=0)
stack_ys_kr_p = np.nan_to_num(stack_ys_kr_p)
stack_ys_kr_p.shape

fig_p1, ax_p1 = plt.subplots(figsize=(axs[1].get_position().height * 20,
                                      axs[1].get_position().width * 14), constrained_layout=Tru
e)

ax_p1.stackplot(df_bo_kr_mY["openYear"], *stack_ys_kr_p,
                ec="none", lw=0.5)
ax_p1.get_children()[20].set_facecolor(c_etc)
ax_p1.set_xlim(1971, 2020)
ax_p1.set_ylim(0, 1)
ax_p1.set_xticks([])
ax_p1.set_yticks([])
ax_p1.spines[["top", "bottom", "left", "right"]].set_visible(False)
fig_p1.savefig("./images/bo_year1.png", dpi=200)
```

In [141]:

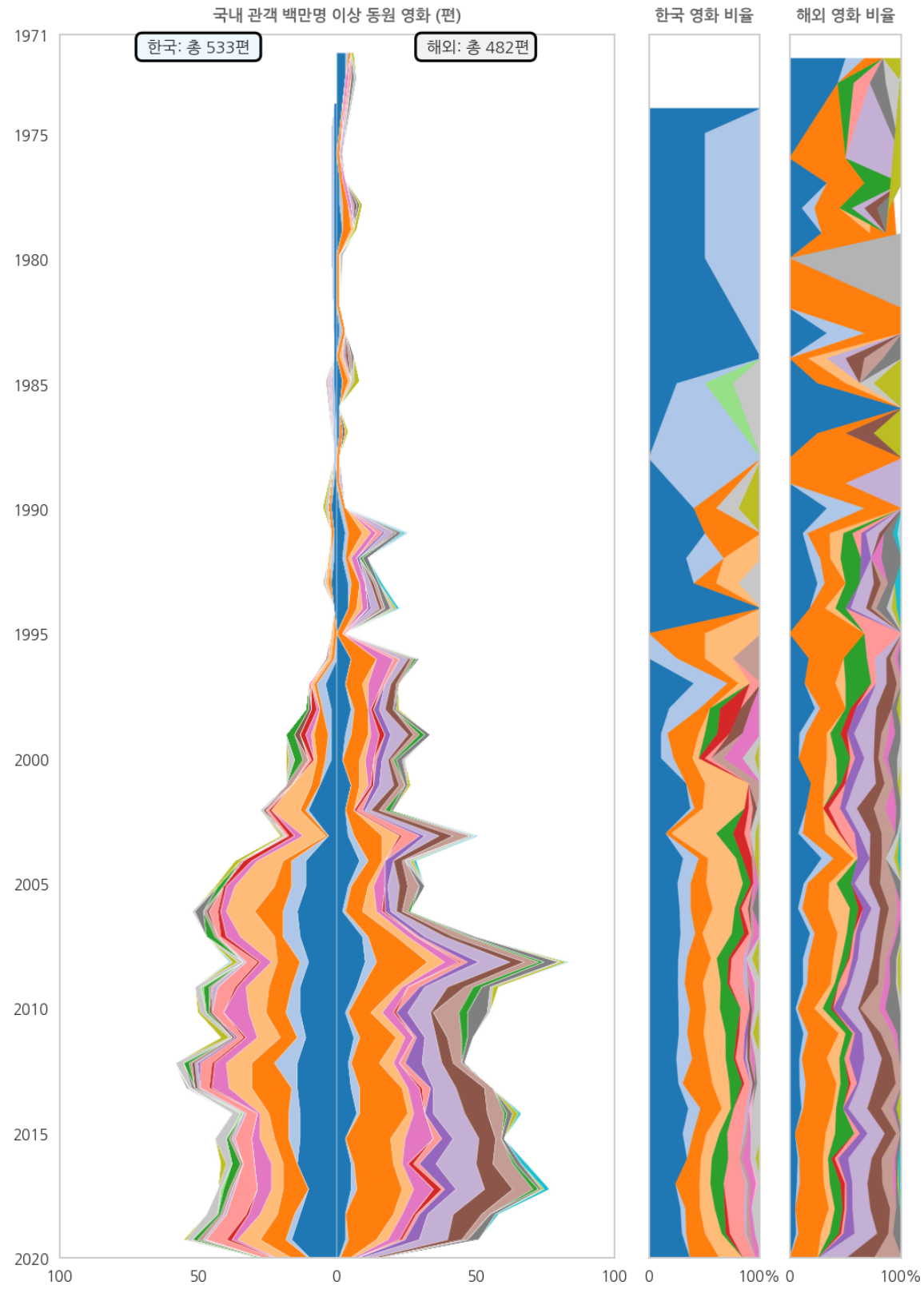
```
# 해외 genre portion

stack_ys_nkr.append(-df_bo_nkr_mY['G_기타'])

stack_ys_nkr_p = np.array(stack_ys_nkr)/np.array(stack_ys_nkr).sum(axis=0)
stack_ys_nkr_p = np.nan_to_num(stack_ys_nkr_p)
stack_ys_nkr_p.shape

fig_p2, ax_p2 = plt.subplots(figsize=(axs[2].get_position().height * 20,
                                      axs[2].get_position().width * 14), constrained_layout=Tru
e)

ax_p2.stackplot(df_bo_nkr_mY["openYear"], *stack_ys_nkr_p,
                ec="none", lw=0.5)
ax_p2.get_children()[20].set_facecolor(c_etc)
ax_p2.set_xlim(1971, 2020)
ax_p2.set_ylim(0, 1)
ax_p2.set_xticks([])
ax_p2.set_yticks([])
ax_p2.spines[["top", "bottom", "left", "right"]].set_visible(False)
fig_p2.savefig("./images/bo_year2.png", dpi=200)
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:5: RuntimeWarning: di
vide by zero encountered in true_divide
  """
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:5: RuntimeWarning: in
valid value encountered in true_divide
  """
```

In [142]:

```
## 조립

# axs[0]
fig0_img = plt.imread("./images/bo_year0.png")
fig0_img = fig0_img.swapaxes(0, 1)[:,::-1, :][10:-10,10:-10,:]
x0, x1 = -100, 100
y1, y0 = 1971, 2020

axs[0].imshow(fig0_img, extent=[x0, x1, y0, y1])
axs[0].set_aspect(abs(x1-x0)/abs(y1-y0)*portion_aspect0*20/14)
axs[0].grid(False)

axs[0].set_xlim(-100, 100)
xticks = range(-100, 150, 50)
axs[0].set_xticks(xticks)
axs[0].set_xticklabels([abs(x) for x in xticks])
yticks = [1971] + list(range(1975, 2025, 5))
axs[0].set_yticks(yticks)
axs[0].set_yticklabels(yticks)

axs[0].set_title("국내 관객 백만명 이상 동원 영화 (편)", fontdict=font_title, pad=16)


axs[0].text(-50, 1971.5, f" 한국: 총 {df_bo_kr_mY['영화편수'].sum()}편 ", ha="center", va="cente
r",
            bbox={"boxstyle":"round", "pad":0.4,
                  "facecolor":"aliceblue", "edgecolor":"k", "linewidth":3},
            zorder=20)

axs[0].text(50, 1971.5, f" 해외: 총 {df_bo_nkr_mY['영화편수'].sum()}편", ha="center", va="cente
r",
            bbox={"boxstyle":"round", "pad":0.4,
                  "facecolor":"0.95", "edgecolor":"k", "linewidth":3},
            zorder=20)

# axs[1] 한국
fig1_img = plt.imread("./images/bo_year1.png")
fig1_img = fig1_img.swapaxes(0, 1)[:,::-1, :][10:-10,10:-10,:]
x0, x1 = 0, 1
y1, y0 = 1971, 2020

axs[1].imshow(fig1_img, extent=[x0, x1, y0, y1])
axs[1].set_aspect(abs(x1-x0)/abs(y1-y0)*portion_aspect1*20/14)
axs[1].grid(False)

xticks = [0, 1]
axs[1].set_xticks(xticks)
axs[1].set_xticklabels([0, "100%"])
axs[1].set_title("한국 영화 비율", fontdict=font_title, pad=16)


# axs[2] 해외
fig2_img = plt.imread("./images/bo_year2.png")
fig2_img = fig2_img.swapaxes(0, 1)[:,::-1, :][10:-10,10:-10,:]
x0, x1 = 0, 1
y1, y0 = 1971, 2020

axs[2].imshow(fig2_img, extent=[x0, x1, y0, y1])
axs[2].set_aspect(abs(x1-x0)/abs(y1-y0)*portion_aspect2*20/14)
```

```
axs[2].grid(False)

xticks = [0, 1]
axs[2].set_xticks(xticks)
axs[2].set_xticklabels([0, "100%"])
axs[2].set_title("해외 영화 비율", fontdict=font_title, pad=16)

display(fig)
```


국내 관객 백만명 이상 동원 영화 (편)
한국 영화 비율
해외 영화 비율
한국: 총 533편
해외: 총 482편
1971
1975
1980
1985
1990
1995
2000
2005
2010
2015
2020
100
50
0
50
100
0
100%
0
100%

In [143]:

```
df_kr_10m_list = df_bo_kr_10m[["영화명", "openYear", "전국 관객수"]].groupby("openYear").agg(list).reset_index()
df_kr_10m_list
```

Out[143]:

	openYear	영화명	전국 관객수
0	2003	[실미도]	[11081000.0]
1	2004	[태극기휘날리며]	[11746135.0]
2	2005	[왕의남자]	[12302831.0]
3	2006	[괴물]	[13019740.0]
4	2009	[해운대]	[11453338.0]
5	2012	[광해,왕이된남자, 도둑들]	[12319542.0, 12983330.0]
6	2013	[변호인, 7번방의선물]	[11372451.0, 12811206.0]
7	2014	[명량, 국제시장]	[17613682.0, 14245998.0]
8	2015	[베테랑, 암살]	[13395400.0, 12705700.0]
9	2016	[부산행]	[11565078.0]
10	2017	[신과함께-죄와벌, 택시운전사]	[14410754.0, 12099971.0]
11	2018	[신과함께-인과연]	[12253247.0]
12	2019	[기생충, 극한직업]	[10313086.0, 16264944.0]

In [144]:

```
df_nkr_10m_list = df_bo_nkr_10m[["영화명", "openYear", "전국 관객수"]].groupby("openYear").agg(list).reset_index()
df_nkr_10m_list
```

Out[144]:

	openYear	영화명	전국 관객수
0	2009	[아바타]	[13624328.0]
1	2014	[인터스텔라, 겨울왕국]	[10273803.0, 10296101.0]
2	2015	[어벤져스:에이지오브울트론]	[10494499.0]
3	2018	[어벤져스:인피니티워]	[11211880.0]
4	2019	[겨울왕국2, 알라딘, 어벤져스:엔드게임]	[13747792.0, 12555894.0, 13934592.0]

In [145]:

```
df_bo_kr_10m[["영화명", "openYear", "전국 관객수"]].sort_values("전국 관객수", ascending=False)
```

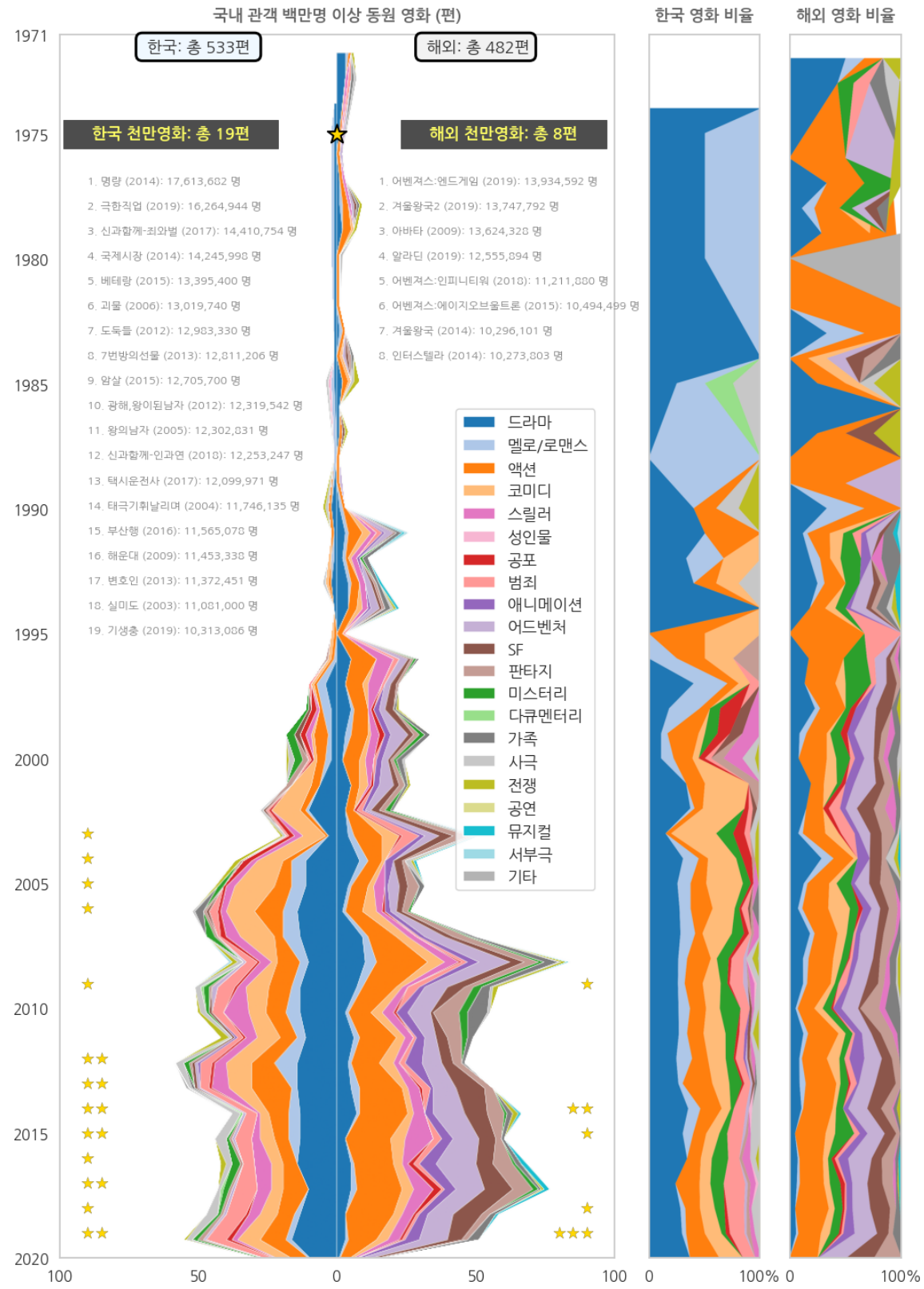
Out[145]:

	영화명	openYear	전국 관객수
4949	명량	2014	17613682.0
1717	극한직업	2019	16264944.0
394	신과함께-죄와벌	2017	14410754.0
7381	국제시장	2014	14245998.0
6879	베테랑	2015	13395400.0
4003	괴물	2006	13019740.0
10522	도둑들	2012	12983330.0
8317	7번방의선물	2013	12811206.0
10124	암살	2015	12705700.0
1210	광해,왕이된남자	2012	12319542.0
7768	왕의남자	2005	12302831.0
393	신과함께-인과연	2018	12253247.0
6987	택시운전사	2017	12099971.0
207	태극기휘날리며	2004	11746135.0
5412	부산행	2016	11565078.0
7380	해운대	2009	11453338.0
7355	변호인	2013	11372451.0
9323	실미도	2003	11081000.0
569	기생충	2019	10313086.0

In [146]:

```
axs[0].scatter(0, 1975, marker="*", s=500, c="gold", ec="k", lw=2, zorder=20)

# 한국 천만영화
axs[0].text(-60, 1975, f"     한국 천만영화: 총 {df_bo_kr_10m.shape[0]}편     ", ha="center",
va="center",
            fontsize="medium", color="#FFFF66", fontweight="bold",
            bbox={"boxstyle":"square", "pad":0.4,
                  "facecolor":"0.3", "edgecolor":"0.3", "linewidth":1},
            zorder=20)

for y in df_kr_10m_list["openYear"].values:
    m_year = df_kr_10m_list.query(f"openYear == {y}")
    n_year = m_year["영화명"].apply(len).values[0]

    for i in range(n_year):
        axs[0].scatter(-1*(90-5*i), y, marker="*", s=200, c="gold", ec="k", lw=0.2)

for i, (_, r) in enumerate(df_bo_kr_10m[["영화명", "openYear", "전국 관객수"]].sort_values("전
국 관객수", ascending=False).iterrows()):
    axs[0].text(-90, 1977+i,
                  f"{i+1}. {r.loc['영화명']} ({r.loc['openYear']}): {format(int(r.loc['전
국 관객수']), ',')} 명",
                  fontsize="x-small", color="gray")

# 해외 천만영화
axs[0].text(60, 1975, f"     해외 천만영화: 총 {df_bo_nkr_10m.shape[0]}편     ", ha="center",
va="center",
            fontsize="medium", color="#FFFF66", fontweight="bold",
            bbox={"boxstyle":"square", "pad":0.4,
                  "facecolor":"0.3", "edgecolor":"0.3", "linewidth":1},
            zorder=20)

for y in df_nkr_10m_list["openYear"].values:
    m_year = df_nkr_10m_list.query(f"openYear == {y}")
    n_year = m_year["영화명"].apply(len).values[0]

    for i in range(n_year):
        axs[0].scatter((90-5*i), y, marker="*", s=200, c="gold", ec="k", lw=0.2)

for i, (_, r) in enumerate(df_bo_nkr_10m[["영화명", "openYear", "전국 관객수"]].sort_values("전
국 관객수", ascending=False).iterrows()):
    axs[0].text(15, 1977+i,
                  f"{i+1}. {r.loc['영화명']} ({r.loc['openYear']}): {format(int(r.loc['전
국 관객수']), ',')} 명",
                  fontsize="x-small", color="gray")

# ### legend
for i, (g, c) in enumerate(zip(genres_order, c_genres)):
    axs[0].bar(0, 1, bottom=1990, label=g, fc=c, zorder=-2)
axs[0].legend(fontsize="medium", bbox_to_anchor=(0.98, 0.7), loc="upper right")

x0, x1 = -100, 100
axs[0].set_aspect(abs(x1-x0)/abs(y1-y0)*portion_aspect0*20/14)
x0, x1 = 0, 1
axs[1].set_aspect(abs(x1-x0)/abs(y1-y0)*portion_aspect1*20/14)
x0, x1 = 0, 1
axs[2].set_aspect(abs(x1-x0)/abs(y1-y0)*portion_aspect2*20/14)
```

```
fig.savefig("./images/bo_year_m.png", dpi=300)
display(fig)
```


국내 관객 백만명 이상 동원 영화 (편)
한국 영화 비율
해외 영화 비율
한국: 총 533편
해외: 총 482편
한국 천만영화: 총 19편
1. 명량 (2014): 17,613,682 명
2. 극한직업 (2019): 16,264,944 명
3. 신과함께-죄와벌 (2017): 14,410,754 명
4. 국제시장 (2014): 14,245,998 명
5. 베테랑 (2015): 13,395,400 명
6. 괴물 (2006): 13,019,740 명
7. 도둑들 (2012): 12,983,330 명
8. 7번방의선물 (2013): 12,811,206 명
9. 암살 (2015): 12,705,700 명
10. 광해,왕이된남자 (2012): 12,319,542 명
11. 왕의남자 (2005): 12,302,831 명
12. 신과함께-인과연 (2018): 12,253,247 명
13. 택시운전사 (2017): 12,099,971 명
14. 태극기휘날리며 (2004): 11,746,135 명
15. 부산행 (2016): 11,565,078 명
16. 해운대 (2009): 11,453,338 명
17. 변호인 (2013): 11,372,451 명
18. 실미도 (2003): 11,081,000 명
19. 기생충 (2019): 10,313,086 명
해외 천만영화: 총 8편
1. 어벤져스:엔드게임 (2019): 13,934,592 명
2. 겨울왕국2 (2019): 13,747,792 명
3. 아바타 (2009): 13,624,328 명
4. 알라딘 (2019): 12,555,894 명
5. 어벤져스:인피니티워 (2018): 11,211,880 명
6. 어벤져스:에이지오브울트론 (2015): 10,494,499 명
7. 겨울왕국 (2014): 10,296,101 명
8. 인터스텔라 (2014): 10,273,803 명
드라마
멜로/로맨스
액션
코미디
스릴러
성인물
공포
범죄
애니메이션
어드벤처
SF
판타지
미스터리
다큐멘터리
가족
사극
전쟁
공연
뮤지컬
서부극
기타
1971
1975
1980
1985
1990
1995
2000
2005
2010
2015
2020
100
50
0
50
100
0
100%
0
100%

In [146]: