데이타 크리닝 연습

In [868]:

```
import pandas as pd

# 판다스 데이터프레임(DataFrame)을 출력할 때, 최대 출력할 수 있는 컬럼을 100개로 늘려줍니다.
# 이렇게 해야 데이터를 분석할 때 출력해서 확인하기 편합니다.
pd.options.display.max_columns = 100
```

데이터 로딩하기

kmong-conversion.csv

먼저 크몽의 로그 데이터를 하나씩 가져오겠습니다. 가장 처음 가져올 데이터는 크몽의 가장 핵심 데이터라고 할 수 있는 `kmong-conversion.csv` 입니다. 여기에는 사용자가 크몽의 웹사이트/모바일 서비스에서 활동한 모든 활동 기록(activity)이 상세하게 담겨져 있으며, 이 정보를 바탕으로 데이터를 분석하여 사용자의 패턴을 유추할 수 있습니다. 컬럼 정보는 다음과 같습니다.

- `eventcategory` : 앱의 이벤트에 대한 카테고리입니다. 크게 다음의 종류가 있습니다.
  - `install` : 앱설치
  - `launch` : 앱실행
  - `deeplinkLaunch` : 딥링크 통한 앱실행
  - `goal` : 앱 내에서의 활동 여부
  - `exit` : 앱 종료
  - `foreground` , background```: 앱을 foreground, background로 전환
  - `launchInSession` : Session의 시작.
- `isfirstactivity` : 해당 이벤트가 해당 유저에 한하여 첫 번째인지 혹은 두 번째 이상인지를 알려주는 값입니다.
  - boolean 형식으로, 해당 이벤트가 해당 유저에 한하여 첫 번째 이벤트이면 True, 아니면 False를 갖습니다.
- `apppackagename` : 앱의 패키지의 고유한 이름입니다. 안드로이드의 경우 applicationId을, iOS의 경우 Bundle ID를 가져오게 됩니다.
- `appversion` : kmong 어플리케이션의 버전입니다.
- `devicetype` : 사용자가 kmong app을 실행하고 있는 기기의 제품명입니다.
- `devicemanufacturer` : 사용자가 kmong app을 실행하고 있는 기기의 제조회사입니다.
- `osversion` : 사용자가 kmong app을 실행하고 있는 기기의 os 버전입니다.
- `canonicaldeviceuuid` : 사용자가 kmong app을 실행하고 있는 기기의 고유 식별자입니다. (사용자를 구분하는 식별자로 이용할 수 있습니다)
- `sourcetype` : 해당 로그를 만든 고객이 유입된 경로입니다. 유료 광고 채널, 바이럴 채널, 앱 마켓 등으로 구분합니다.
- `channel` : sourcetype과 동일하지만 조금 더 세분화되어 있습니다.
  - unattributed, WEB, google-play, m_naver, google, (not set), google.adwords, m_naverpowercontents, pc_naver, apple.searchads, facebook, m_daum
- `params_campaign` : 마케터가 입력한 캠페인 파라미터 중 캠페인명입니다.
- `params_medium` : 마케터가 입력한 캠페인 파라미터 중 매체입니다.
- `params_term` : 마케터가 입력한 캠페인 파라미터 중 키워드입니다.
- `inappeventcategory` : 인앱이벤트 식별자값으로 인앱이벤트를 지칭하는 3개의 값(category > action > label) 중 가장 상위에 속하는 구분값입니다.
  - `eventcategory` 가 goal인 경우에만 제공됩니다.
  - 차후에 읽어올 funnel 데이터셋과 연결할 수 있습니다.

    - ex) seller_selling_history.view, gig_detail.view
- `inappeventlabel` : categoryid와 동일합니다.
    - 차후에 읽어올 category 데이터셋과 연결할 수 있습니다.
    - 인앱이벤트를 지칭하는 3개의 값(category > action > label) 중 마지막에 속하는 구분값입니다.
- `eventdatetime` : 모바일 클라이언트 상에서의 실제 이벤트 발생시간입니다. ISO 8601 표준에 따라서 제공됩니다.
- `isfirstgoalactivity` : goal 이벤트의 경우, 해당 Goal Label, Description, Key, Category를 가진 이벤트가 해당 유저에 한하여 첫 번째인지 혹은 두 번째 이상 인지를 알려주는 boolean 값입니다. 예를 들어 동일한 유저로부터 두 번 이상의 같은 Goal 이벤트가 발생한 경우(e.g. 동일한 물건 재구매 등) 이 값은 false가 됩니다.
- `event_rank` : 데이터를 정렬하는데 필요한 컬럼입니다. log엔 기록되지 않으며, DS_School에서 제공하는 전처리된 컬럼입니다.

In [869]:

```
raw_log = pd.read_csv("./data/kmong-conversion.csv")

print(raw_log.shape)

raw_log.head()
```

```
(434244, 19)
```

Out[869]:

	eventcategory	isfirstactivity	apppackagename	appversion	devicetype	devicemanufacturer
0	goal	False	com.kmong.iOS	4.0.4	iPhone	Apple
1	goal	False	com.kmong.kmong	3.3.5	SM-N935S	samsung
2	goal	False	com.kmong.iOS	4.0.4	iPhone	Apple
3	foreground	NaN	com.kmong.iOS	4.0.4	iPhone	Apple
4	goal	False	com.kmong.iOS	4.0.4	iPhone	Apple

In [870]:

```
dir(raw_log.columns)
```

```
 'isnull',
 'item',
 'join',
 'map',
 'max',
 'memory_usage',
 'min',
 'name',
 'names',
 'nbytes',
 'ndim',
 'nlevels',
 'notna',
 'notnull',
 'nunique',
 'putmask',
 'ravel',
 'reindex',
 'rename',
```

kmong-funnel.csv

그 다음 가져올 데이터셋은 퍼널(Funnel) 정보를 나타내는 `kmong-funnel.csv` 입니다. 퍼널은 사용자가 서비스에 접속한 뒤 상품을 구매하기까지의 모든 경로를 구조화 한 것을 의미합니다.

사용자가 특정 상품을 결제하기 위해선, 해당 상품의 설명을 적은 페이지를 방문해야 하며, 상품 설명 페이지를 방문하기 위해서는 해당 상품이 리스트되어있는 페이지를 방문해야 합니다. 이런 방식으로 웹사이트를 설명하는 것을 일명 퍼널이라고 합니다.

이 퍼널 정보를 이용해 사용자가 상품을 구매하는 전환(Conversion) 여부, 내지는 서비스를 이탈하는 이탈(Churn) 여부를 추적할 수 있으며, 이를 통해서 웹사이트나 모바일서비스를 개선하여 전환율을 높이고 이탈률을 낮출 수 있습니다. 크몽에서는 서비스에서 사용하는 퍼널에 대한 설명을 `kmong-funnel.csv` 파일에 저장해두고 있습니다.

In [871]:

```
raw_funnel = pd.read_csv("./data/kmong-funnel.csv")

print(raw_funnel.shape)

raw_funnel
```

(53, 6)

Out[871]:

	Lv2	viewid	viewid desc	Lv1	funnel name	funnel desc
0	1100	home	홈 (탭)	11	home	홈
1	1210	category_list	카테고리 목록 (탭)	12	category	카테고리
2	1200	category_gig	카테고리-상품목록	12	category	카테고리
3	1300	search	검색	13	search	검색
4	1301	search_gig	검색-상품목록	13	search	검색
5	1302	search_seller	검색-전문가	13	search	검색
6	1400	gig_detail	상품상세	14	gig	상품
7	1401	gig_detail_option	상품상세-상품선택	14	gig	상품
8	1420	profile	전문가프로필	14	gig	상품
9	1520	login_sns	간편로그인	15	login	로그인
10	1500	login_email	간편로그인-이메일로그인	15	login	로그인
11	1510	find_account	간편로그인-이메일로그인-아이디/비밀번호-찾기	15	login	로그인
12	1600	inbox	메시지목록 (탭)	16	inbox	메시지
13	1610	inbox_detail	메시지목록-상세	16	inbox	메시지
14	1611	inbox_detail_orderRequest	메시지목록-상세-결제요청	16	inbox	메시지
15	1700	bookmarks_gig	메뉴목록-찜목록-서비스	17	bookmarks	찜하기
16	1701	bookmarks_portfolio	메뉴목록-찜목록-포트폴리오	17	bookmarks	찜하기
17	1801	order_addOption	상품상세-주문하기	18	order	결제
18	1800	order_payment	상품상세-주문하기-결제하기	18	order	결제
19	1810	kmongCash_charge	메뉴목록-크몽캐시-캐시충전하기	18	order	결제
20	1900	thankyou	결제완료	19	thankyou	결제완료

	Lv2	viewid	viewid desc	Lv1	funnel name	funnel desc
21	2000	signup	개인/기업-회원가입	20	new	신규등록
22	2001	mobileAuth	휴대폰인증	20	new	신규등록
23	2020	seller_gig_register	메뉴목록-서비스등록	20	new	신규등록
24	2200	account_setting	메뉴목록-계정설정	22	account	계정관리 - 회원공통
25	2210	account_setting_notification	메뉴목록-계정설정-알림설정	22	account	계정관리 - 회원공통
26	2220	account_setting_AuthStatus	메뉴목록-계정설정-인증정보	22	account	계정관리 - 회원공통
27	2230	account_setting_pwd	메뉴목록-계정설정-비밀번호 변경	22	account	계정관리 - 회원공통
28	2240	account_setting_profile	메뉴목록-계정설정-프로필	22	account	계정관리 - 회원공통
29	2320	seller_gig	메뉴목록-나의서비스	23	account_seller	계정관리 - 판매자
30	2400	menu	메뉴목록 (탭)	24	transaction_history	거래관리
31	2410	buyer_order_track	메뉴목록-구매관리-거래메시지	24	transaction_history	거래관리
32	2411	buyer_order_track_cancel	메뉴목록-구매관리-거래메시지-취소/문제해결	24	transaction_history	거래관리
33	2412	buyer_order_track_cancel_submit	메뉴목록-구매관리-거래메시지-취소/문제해결-제출	24	transaction_history	거래관리
34	2420	buyer_order_history	메뉴목록-구매관리	24	transaction_history	거래관리
35	2421	buyer_coupon_history	메뉴목록-쿠폰내역	24	transaction_history	거래관리
36	2430	seller_selling_history	메뉴목록-판매관리	24	transaction_history	거래관리
37	2431	seller_mileage_history	메뉴목록-마일리지 내역	24	transaction_history	거래관리
38	2432	seller_profits_history	메뉴목록-수익관리	24	transaction_history	거래관리
39	2450	buyer_payment_history	메뉴목록-결제내역	24	transaction_history	거래관리

	Lv2	viewid	viewid desc	Lv1	funnel name	funnel desc
40	2451	kmongCash_history	메뉴목록-크몽캐시	24	transaction_history	거래관리
41	2452	kmongCash_history_refundRequest	메뉴목록-크몽캐시-캐시환불	24	transaction_history	거래관리
42	2510	about_howToUse	메뉴목록-크몽이용방법	25	info	소개 및 안내
43	2520	termsOfService	메인하단-이용약관	25	info	소개 및 안내
44	2521	privacyPolicy	메인하단-개인정보처리방침	25	info	소개 및 안내
45	2530	tutorial	튜토리얼	25	info	소개 및 안내
46	2700	referrerProgram	메뉴목록-친구초대	27	referral_program	친구초대프로그램
47	2800	event	이벤트	28	event	이벤트 프로모션
48	3010	kmongNotification	메뉴목록-크몽안내	30	notification	알림
49	3011	kmongNotification_detail	메뉴목록-크몽안내-상세	30	notification	알림
50	3000	notification	알림-알림	30	notification	알림
51	3001	notification_detail	알림-알림-상세	30	notification	알림
52	9910	NPS	NPS	99	others	기타

kmong-category.csv

마지막으로 가져올 데이터셋은 크몽에서 판매하는 상품의 카테고리를 나타내는 `kmong-category.csv` 입니다. 크몽에서는 디자인, 번역, 컨텐츠 제작등 다양한 상품을 판매하고 있는데, 이 상품을 그룹화하고 정리할 수 있도록 만든 것이 카테고리입니다. 크몽에서는 이 값을 `kmong-category.csv` 라는 변수에 저장해두고 있습니다.

In [872]:

```
raw_category = pd.read_csv("./data/kmong-category.csv")


print(raw_category.shape)

raw_category
```

(245, 9)

Out[872]:

	depth	categoryid	categoryname	cat1_id	cat2_id	cat3_id	cat1	cat2	cat3
0	1	1	디자인	1	NaN	NaN	디자인	NaN	NaN
1	1	2	마케팅	2	NaN	NaN	마케팅	NaN	NaN
2	1	3	번역·통역	3	NaN	NaN	번역·통역	NaN	NaN
3	1	4	문서작성	4	NaN	NaN	문서작성	NaN	NaN
4	1	6	IT·프로그래밍	6	NaN	NaN	IT·프로그래밍	NaN	NaN
...	...	...	...	...	...	...	...	...	...
240	3	72501	모델	7	725.0	72501.0	콘텐츠 제작	엔터테이너	모델
241	3	72502	MC	7	725.0	72502.0	콘텐츠 제작	엔터테이너	MC
242	3	72503	배우	7	725.0	72503.0	콘텐츠 제작	엔터테이너	배우
243	3	72504	공연	7	725.0	72504.0	콘텐츠 제작	엔터테이너	공연
244	3	72505	기타	7	725.0	72505.0	콘텐츠 제작	엔터테이너	기타

245 rows × 9 columns

데이터 정리하기

먼저 가장 기본이 되는 kmong-conversion.csv 부터 정리하겠습니다. 이 데이터에는 사용자의 활동 기록을 자세하게 정리하였다는 장점이 있지만, 데이터가 다소 지저분하게 기록되어 있기 때문에, 이 데이터를 정리하지 않은 채 분석하는 것은 많은 시간을 낭비하게 될 것 같습니다. 그러므로 이번 기회에 데이터를 깔끔하게 정리하여 저장하도록 하겠습니다.

1. canonicaldeviceuuid 컬럼명을 userid 컬럼명으로 바꿔주세요.

먼저 canonicaldeviceuuid 라는 컬럼명을 바꾸고 싶습니다. 이 컬럼은 크몽을 이용하는 한 명의 사용자를 나타내는 중요한 정보임에는 틀림없지만, canonicaldeviceuuid 라는 표현 자체가 크몽, 내지는 다른 몇몇 분야에서만 쓰이는 표현이기 때문에 이 데이터를 처음 보는 사람의 입장에서는 직관적으로 이해하기 힘듭니다. 그렇기 때문에 이 컬럼의 이름을 많은 사람들이 직관적으로 이해할 수 있는 userid 로 바꾸고 싶습니다. 최종적으로 다음의 결과가 나와야 합니다.

canonicaldeviceuuid	userid

	canonicaldeviceuuid	userid
0	F36FAA62-ADAC-4AA5-9B00-1FD6CB7EE957	F36FAA62-ADAC-4AA5-9B00-1FD6CB7EE957
1	8a871e50-0717-4aed-9bad-04ac3c3793be	8a871e50-0717-4aed-9bad-04ac3c3793be
2	A9E5778A-8F3D-4597-9718-74BF953A9F64	A9E5778A-8F3D-4597-9718-74BF953A9F64
3	168761CB-CB67-4592-867D-52780D651297	168761CB-CB67-4592-867D-52780D651297
4	ACABB7C0-4C76-413A-B314-E5D6DA0D0E5D	ACABB7C0-4C76-413A-B314-E5D6DA0D0E5D

In [873]:

```
# Write your code here!

print(raw_log.columns) # 컬럼 이름 바꾸기 전

raw_log.rename(columns={"canonicaldeviceuuid":"userid"}, inplace = True)

print(raw_log.columns) # 컬럼 이름 바꾼 후
```

```
Index(['eventcategory', 'isfirstactivity', 'apppackagename', 'appversion',
       'devicetype', 'devicemanufacturer', 'osversion', 'canonicaldeviceuuid',
       'sourcetype', 'channel', 'params_campaign', 'params_medium',
       'params_term', 'inappeventcategory', 'inappeventlabel', 'eventdatetime',
       'rowuuid', 'isfirstgoalactivity', 'event_rank'],
      dtype='object')
Index(['eventcategory', 'isfirstactivity', 'apppackagename', 'appversion',
       'devicetype', 'devicemanufacturer', 'osversion', 'userid', 'sourcetype',
       'channel', 'params_campaign', 'params_medium', 'params_term',
       'inappeventcategory', 'inappeventlabel', 'eventdatetime', 'rowuuid',
       'isfirstgoalactivity', 'event_rank'],
      dtype='object')
```

2. eventdatetime 컬럼은 datetime 형태로 바꾼 뒤 연/월/일/시/분/초 데이터를 추출해주세요.

eventdatetime 은 날짜와 시간 형식으로 표현되어 있습니다. (ex: 2018-01-01T00:00:00+09:00) 하지만 이 데이터를 처음 읽을 때, 판다스는 날짜 데이터로 인식하지 않고 문자열(object, 내지는 string)으로 인식합니다. 이 데이터를 날짜와 시간 형식을 나타내는 datetime 형태로 바꾼 뒤, 연/월/일/시/분/초 데이터를 추출해주세요. 최종적으로 다음의 결과가 나와야 합니다.

	eventdatetime	eventdatetime_year	eventdatetime_month	eventdatetime_day	eventdatetime_hour	eventdatetim
0	2018-09-27 15:00:00	2018	9	27	15	
1	2018-09-27 15:00:00	2018	9	27	15	
2	2018-09-27 15:00:00	2018	9	27	15	
3	2018-09-27 15:00:01	2018	9	27	15	
4	2018-09-27 15:00:02	2018	9	27	15	

In [874]:

```
# Write your code here!
from datetime import datetime

#raw_log['eventdatetime']=pd.to_datetime(raw_log['eventdatetime'], format='%Y-%m-%dT%H:%M:%S+09:00')
raw_log['eventdatetime']=pd.to_datetime(raw_log['eventdatetime'])
raw_log['eventdatetime_year']=raw_log['eventdatetime'].dt.year
raw_log['eventdatetime_month']=raw_log['eventdatetime'].dt.month
raw_log['eventdatetime_day']=raw_log['eventdatetime'].dt.day
raw_log['eventdatetime_hour']=raw_log['eventdatetime'].dt.hour
raw_log['eventdatetime_minute']=raw_log['eventdatetime'].dt.minute
raw_log['eventdatetime_second']=raw_log['eventdatetime'].dt.second
raw_log.head()
```

Out[874]:

	eventcategory	isfirstactivity	apppackagename	appversion	devicetype	devicemanufacturer
0	goal	False	com.kmong.iOS	4.0.4	iPhone	Apple
1	goal	False	com.kmong.kmong	3.3.5	SM-N935S	samsung
2	goal	False	com.kmong.iOS	4.0.4	iPhone	Apple
3	foreground	NaN	com.kmong.iOS	4.0.4	iPhone	Apple
4	goal	False	com.kmong.iOS	4.0.4	iPhone	Apple

3. osversion 컬럼에 들어가 있는 정보를 여러 개의 컬럼으로 나눠주세요.

osversion 컬럼을 사용자가 현재 사용하고 있는 스마트폰 디바이스의 운영체제와 그 버전을 나타냅니다. 여기에는 크게 1) 현재 사용하고 있는 운영체제(Operating System, 이하 OS), 2) 그리고 해당 운영체제의 버전(version) 정보가 기록되어 있습니다. 가령 osversion 에 들어있는 값이 iOS11.4.1이라면, 이 스마트폰의 운영체제는 iOS(아이폰), 운영체제의 버전은 11.4.1 이라는 사실을 알 수 있습니다.

이 데이터를 차후 데이터 분석용으로 다루기 쉽게, ostype(clean) 과 osversion(clean) 이라는 두 개의 컬럼으로 분리하고 싶습니다. 가령 에를 들자면

- osversion 컬럼값이 iOS11.4.1이라면, 새로운 컬럼인 ostype(clean) 에는 iOS가, osversion(clean) 에는 11.4.1이 들어가야 합니다.
- osversion 컬럼값이 Android7.0이라면, 새로운 컬럼인 ostype(clean) 에는 Android가, osversion(clean) 에는 7.0이 들어가야 합니다.

최종적으로는 다음의 결과가 나와야 합니다.

	osversion osversion	ostype(clean) ostype(clean)	osversion(clean) osversion(clean)
0	iOS11.4.1	iOS	11.4.1
1	Android7.0	Android	7.0
2	iOS12.0	iOS	12.0
3	iOS11.4.1	iOS	11.4.1
4	iOS11.4.1	iOS	11.4.1

In [875]:

```
# Write your code here!
import re
# raw_log['osversion']
# raw_log.groupby('osversion').count() # ostype의 종류는 iOS와 Android

def gettype(osversion):
    if re.search('iOS', osversion):
        return 'iOS'
    if re.search('Android', osversion):
        return 'Android'

def getversion(osversion):
    if re.search('iOS', osversion):
        return osversion[3:]
    if re.search('Android', osversion):
        return osversion[7:]

raw_log['ostype(clean)'] = raw_log['osversion'].apply(gettype)
raw_log['osversion(clean)'] = raw_log['osversion'].apply(getversion)
```

In [876]:

```
raw_log.head()
```

Out[876]:

	eventcategory	isfirstactivity	apppackagename	appversion	devicetype	devicemanufacturer
0	goal	False	com.kmong.iOS	4.0.4	iPhone	Apple
1	goal	False	com.kmong.kmong	3.3.5	SM-N935S	samsung
2	goal	False	com.kmong.iOS	4.0.4	iPhone	Apple
3	foreground	NaN	com.kmong.iOS	4.0.4	iPhone	Apple
4	goal	False	com.kmong.iOS	4.0.4	iPhone	Apple

4. devicemanufacturer 컬럼 정보를 정리해주세요.

devicemanufacturer 컬럼은 사용자가 현재 사용하고 있는 스마트폰 디바이스를 제조한 제조사의 정보가 들어가 있습니다. 가령 samsung 이라고 적혀있으면 이 스마트폰은 삼성전자가 제조했다고 볼 수 있고, LG Electronics 라고 적혀있으면 이 스마트폰은 LG전자가 제조했다고 볼 수 있습니다.

하지만 devicemanufacturer 컬럼에는 이 외에도 샤오미(Xiaomi), 폭스콘(Foxconn), 팬텍(PANTECH), 화웨이(HUAWEI)와 같은 마이너한 제조사 정보도 들어가 있습니다. 이 제조사들은 전체 사용자의 2%도 되지 않기 때문에, 이 데이터를 무시하거나 하나의 값(ex: Others)으로 통일해주면 데이터 분석가가 더 효율적으로 데이터를 분석할 수 있을 것 같습니다.

그러므로 devicemanufacturer 컬럼의 값을 정리해보겠습니다. 정리 방식은 다음과 같습니다.

- samsung 이라고 적혀있는 값은 앞 글자를 대문자로 바꿉니다. (= Samsung)
- LGE 와 LG Electronics 라는 값은 LG 로 통일합니다.
- Apple , Samsung , LG 를 제외한 나머지는 Other 로 묶습니다.

최종적으로는 다음의 결과가 나와야 합니다.

- Samsung - 230313
- Apple - 163566
- LG - 32649
- Others - 7716

In [877]:

```
# Write your code here!

def func4(device):
    if device == "samsung":
        return 'Samsung'
    elif device == "LGE" or device == "LG Electronics":
        return 'LG'
    elif device == "Apple":
        return "Apple"
    else:
        return 'Others'
raw_log['devicemanufacturer'] = raw_log['devicemanufacturer'].apply(func4)
raw_log.groupby('devicemanufacturer').count()
```

Out[877]:

devicemanufacturer	eventcategory	isfirstactivity	apppackagename	appversion	devicetype	osv
Apple	163566	117204	163566	163566	163566	
LG	32649	26121	32649	32649	32649	
Others	7716	6338	7716	7716	7716	
Samsung	230313	181502	230313	230313	230313	

5. channel 컬럼 정보를 정리해주세요.

channel 컬럼을 사용자가 크몽 서비스에 유입된 경로를 기록한 정보입니다. 여기에는 웹(WEB), 구글(ex: google-play, google, google.adwords, etc), 네이버(ex: m_naver, m_naverpowercontents, pc_naver)등이 기록되어 있습니다. 이 정보도 마찬가지로 데이터를 분석하기 어렵게 산재되어 있기 때문에, 깔끔하게 정리해주고 싶습니다. 정리 방식은 다음과 같습니다.

- 컬럼값에 google이라고 들어간 값(ex: google-play, google.adwords, etc)은 전부 google로 통일합니다.
- 컬럼값에 daum이라고 들어간 값(ex: m_daum, etc)은 전부 daum으로 통일합니다.
- 컬럼값이 naver라고 들어간 값(ex: m_naver, pc_naver, m_naverpowercontents, etc)은 전부 naver로 통일합니다.
- 컬럼값에 apple이라고 들어간 값(ex: apple.searchads, etc)은 전부 apple로 통일합니다.
- 컬럼값에 WEB이라고 들어간 값(ex: WEB)은 전부 web으로 통일합니다.

최종적으로는 다음의 결과가 나와야 합니다.

- unattributed - 270834
- web - 27639
- google - 27235
- (not set) - 2444
- naver - 1224
- daum - 966
- apple - 697
- facebook - 126

In [878]:

```
# Write your code here!
# raw_log.groupby('channel').count() # 확인

def func5(channel):
    if 'naver' in str(channel):
        return 'naver'
    elif 'google' in str(channel):
        return 'google'
    elif 'daum' in str(channel):
        return 'daum'
    elif 'apple' in str(channel):
        return 'apple'
    elif 'Web' in str(channel):
        return 'web'
    elif 'facebook' in str(channel):
        return 'facebook'
#     elif 'unattributed' in str(channel): # 이 부분 없어도 정상적으로 되는데 왜 되는지 모름
#         return 'unattributed'
    else:
        return channel

raw_log['channel'] = raw_log['channel'].apply(func5)

#raw_log.groupby('channel').count() # 확인
raw_log['channel'].value_counts() # 확인2
```

Out[878]:

```
unattributed    270834
WEB              27639
google           27235
(not set)         2444
naver             1224
daum               966
apple              697
facebook           126
Name: channel, dtype: int64
```

6. inappeventcategory 컬럼에 들어가 있는 정보를 여러 개의 컬럼으로 나눠주세요.

inappeventcategory 에는 사용자 액티비티를 나타내는 정보가 들어있습니다. 크몽에 방문한 고객이 상품 페이지를 보고 있는지, 구매를 진행중인지 등에 대한 정보가 이 컬럼에 담겨있다고 보시면 됩니다. 또한 inappeventcategory 에 있는 정보는 차후 퍼널(funnel) 데이터를 합치는데 사용되기도 합니다. 이 데이터를 정리해주세요. 정리 방식은 다음과 같습니다.

- viewcategory - inappeventcategory 에서 언더바(_)를 기준으로 왼쪽 텍스트만 가져옵니다.
- viewid - inappeventcategory 에서 점(.)의 왼쪽 텍스트만 가져옵니다.
- viewaction - inappeventcategory 에서 점(.)의 오른쪽 텍스트만 가져옵니다.

최종적으로는 다음의 결과가 나와야 합니다.

	inappeventcategory	viewcategory	viewid	viewaction
0	home.view	home	home	view
1	gig_detail.view	gig	gig_detail	view
2	inbox_detail.view	inbox	inbox_detail	view

	inappeventcategory	viewcategory	viewid	viewaction
3	NaN	NaN	NaN	NaN
4	buyer_order_track.view	buyer	buyer_order_track	view

In [879]:

```
# Write your code here!
raw_log.groupby('inappeventcategory').count()
def getviewcategory(inappeventcategory):
    idx = str(inappeventcategory).find('_')
    idx2 = str(inappeventcategory).find('.')
    if idx == -1:
        if idx2 == -1:
            return inappeventcategory
        else:
            return inappeventcategory[:idx2]
    else:
        return inappeventcategory[:idx]
def getviewid(inappeventcategory):
    idx = str(inappeventcategory).find('.')
    if idx == -1:
        return inappeventcategory
    else:
        return inappeventcategory[:idx]
def getviewaction(inappeventcategory):
    idx = str(inappeventcategory).find('.')
    if idx == -1:
        return inappeventcategory
    else:
        return inappeventcategory[(idx+1):]
raw_log['viewcategory'] = raw_log['inappeventcategory'].apply(getviewcategory)
raw_log['viewid'] = raw_log['inappeventcategory'].apply(getviewid)
raw_log['viewaction'] = raw_log['inappeventcategory'].apply(getviewaction)
raw_log.head()

# 방법2
# raw_log1['raw_log']=raw_log['inappeventcategory'].str.split('_').str[0]
# raw_log['viewcategory']=raw_log1['raw_log'].str.split('.').str[0]
# raw_log['viewid']=raw_log['inappeventcategory'].str.split('.').str[0]
# raw_log['viewaction']=raw_log['inappeventcategory'].str.split('.').str[1]

# raw_log.tail()
```

Out[879]:

	eventcategory	isfirstactivity	apppackagename	appversion	devicetype	devicemanufacture
0	goal	False	com.kmong.iOS	4.0.4	iPhone	Appl
1	goal	False	com.kmong.kmong	3.3.5	SM-N935S	Samsun
2	goal	False	com.kmong.iOS	4.0.4	iPhone	Appl
3	foreground	NaN	com.kmong.iOS	4.0.4	iPhone	Appl

	eventcategory	isfirstactivity	apppackagename	appversion	devicetype	devicemanufacture
4	goal	False	com.kmong.iOS	4.0.4	iPhone	Appl

7. 필요하지 않은 컬럼을 버려주세요.

여기까지 했으면 대부분의 컬럼이 잘 정리된 것 같습니다. 이제 남은 것은 필요하지 않은 컬럼을 버리고, 기존 컬럼의 이름을 직관적으로 수정하는 일들만 남았습니다. 먼저 필요하지 않은 컬럼을 제거하겠습니다. 다음의 컬럼을 데이터에서 제거해주세요.

- `osversion` - 이제 이 컬럼을 버린 뒤 `osversion(clean)` 을 사용할 것입니다.
- `devicemanufacturer` - 이제 이 컬럼을 버린 뒤 `devicemanufacturer(clean)` 을 사용할 것입니다.
- `canonicaldeviceuuid` - 이제 이 컬럼을 버린 뒤 `userid` 를 사용할 것입니다.
- `channel` - 이제 이 컬럼을 버린 뒤 `channel(clean)` 을 사용할 것입니다.
- `event_rank` - 이 컬럼은 애초부터 사용할 필요가 없기 때문에 제거해줘도 됩니다.

최종적으로는 다음의 결과가 나와야 합니다.

	sourcetype	params_campaign	params_medium	params_term	inappeventcategory	inappeventlabel	eventdateti
	unattributed	NaN	NaN	NaN	home.view	NaN	2018-09 15:00
	unattributed	NaN	NaN	NaN	gig_detail.view	41201.0	2018-09 15:00
	unattributed	NaN	NaN	NaN	inbox_detail.view	NaN	2018-09 15:00
	NaN	NaN	NaN	NaN	NaN	NaN	2018-09 15:00
	viral	NaN	NaN	NaN	buyer_order_track.view	NaN	2018-09 15:00

In [880]:

```
raw_log = raw_log.drop(['osversion','event_rank'], axis=1)
raw_log.rename(columns={'channel':'channel(clean)'},inplace=True)
raw_log.rename(columns={'devicemanufacturer':'devicemanufacturer(clean)'},inplace=True)

# 어디다 쓰는건지 모름
# new_columns = ['eventcategory', 'isfirstactivity', 'apppackagename', 'appversion',
#                'devicetype', 'sourcetype', 'params_campaign', 'params_medium', 'params_term',
#                'inappeventcategory', 'inappeventlabel', 'eventdatetime',
#                'rowuuid', 'isfirstgoalactivity', 'ostype(clean)',
#                'osversion(clean)', 'devicemanufacturer(clean)', 'userid', 'channel(clean)',
#                'eventdatetime_year', 'eventdatetime_month', 'eventdatetime_day',
#                'eventdatetime_hour', 'eventdatetime_minute', 'eventdatetime_second',
#                'viewid', 'viewaction', 'viewcategory']
# raw_log.columns = new_columns

raw_log.head()
```

Out[880]:

	eventcategory	isfirstactivity	apppackagename	appversion	devicetype	devicemanufacturer(c
0	goal	False	com.kmong.iOS	4.0.4	iPhone	A
1	goal	False	com.kmong.kmong	3.3.5	SM-N935S	Sam
2	goal	False	com.kmong.iOS	4.0.4	iPhone	A
3	foreground	NaN	com.kmong.iOS	4.0.4	iPhone	A
4	goal	False	com.kmong.iOS	4.0.4	iPhone	A

8. 이름을 통일성있게 변경해주세요.

데이터를 분석할 때 컬럼명의 형식을 동일하게 유지하는 것이 중요합니다. 가령 회사 내부 데이터 팀에서 컬럼에서 두 개의 단어를 하이픈(-)으로 연결하기로 합의했다면 모든 컬럼을 하이픈으로 연결해야 하고, 언더바(_)로 연결하기로 합의했다면 모든 컬럼을 언더바로 연결해야 합니다.

하지만 아쉽게도, 현재 데이터의 컬럼명에는 이런 통일성이 잘 갖춰져 있는 것 같습니다. (ex: `eventcategory` vs `params_term`) 그러므로 데이터의 컬럼명을 작성하는 형식을 하나로 통일하도록 하겠습니다. 이번 데이터에서 컬럼명을 짓는 방식은 다음과 같습니다.

- 단어와 단어 사이는 언제나 언더바(_)로 구분합니다. 가령 하이픈(-)으로 구분하거나, 띄어쓰기를 사용하면 안 됩니다. (ex: event_category는 허용하지만, eventcategory나 event-category, 또는 event category는 허용하지 않습니다)
- 컬럼명에 (clean)이라고 들어간 표현은 이제는 지워줘도 될 것 같습니다. (ex channel(clean) -> channel)

최종적으로는 다음의 결과가 나와야 합니다.

	event_category	is_first_activity	app_package_name	app_version	device_type	source_type	params_campaig
0	goal	False	com.kmong.iOS	4.0.4	iPhone	unattributed	Nal
1	goal	False	com.kmong.kmong	3.3.5	SM-N935S	unattributed	Nal
2	goal	False	com.kmong.iOS	4.0.4	iPhone	unattributed	Nal
3	foreground	NaN	com.kmong.iOS	4.0.4	iPhone	NaN	Nal
4	goal	False	com.kmong.iOS	4.0.4	iPhone	viral	Nal

In [881]:

```
# Write your code here!
# 위에서 명시한 형식에 맞게 컬럼명을 새롭게 변경합니다.
# 이를 new_columns라는 이름의 변수에 파이썬 리스트로 집어넣습니다.

#정렬
new_columns = ['eventcategory', 'isfirstactivity', 'apppackagename', 'appversion',
               'devicetype', 'sourcetype', 'params_campaign', 'params_medium', 'params_term',
               'inappeventcategory', 'inappeventlabel', 'eventdatetime',
               'rowuuid', 'isfirstgoalactivity', 'ostype(clean)',
               'osversion(clean)', 'devicemanufacturer(clean)', 'userid', 'channel(clean)',
               'eventdatetime_year', 'eventdatetime_month', 'eventdatetime_day',
               'eventdatetime_hour', 'eventdatetime_minute', 'eventdatetime_second',
               'viewid', 'viewaction', 'viewcategory']

raw_log = raw_log.reindex(columns=new_columns)

new_columns = ['event_category', 'is_first_activity', 'app_package_name', 'app_version',
               'device_type', 'source_type', 'params_campaign', 'params_medium', 'params_term',
               'in_app_event_category', 'in_app_event_label', 'event_datetime',
               'row_uuid', 'is_first_goal_activity', 'os_type',
               'os_version', 'device_manufacturer', 'user_id', 'channel',
               'event_datetime_year', 'event_datetime_month', 'event_datetime_day',
               'event_datetime_hour', 'event_datetime_minute', 'event_datetime_second',
               'view_id', 'view_action', 'view_category']

raw_log.columns = new_columns

raw_log
```

Out[881]:

	event_category	is_first_activity	app_package_name	app_version	device_type	sourc
0	goal	False	com.kmong.iOS	4.0.4	iPhone	unat
1	goal	False	com.kmong.kmong	3.3.5	SM-N935S	unat
2	goal	False	com.kmong.iOS	4.0.4	iPhone	unat
3	foreground	NaN	com.kmong.iOS	4.0.4	iPhone	
4	goal	False	com.kmong.iOS	4.0.4	iPhone	
...	...	...	...	...	...	
434239	goal	False	com.kmong.kmong	3.3.5	SM-G930L	s

	event_category	is_first_activity	app_package_name	app_version	device_type	sourc
434240	goal	False	com.kmong.kmong	3.3.5	SM-N950N	unat
434241	goal	False	com.kmong.kmong	3.3.5	SM-G930L	s
434242	exit	NaN	com.kmong.kmong	3.3.5	SM-N950N	
434243	goal	False	com.kmong.kmong	3.3.5	SM-G965N	unat

434244 rows × 28 columns

9. 컬럼을 정렬해주세요.

마지막은 컬럼을 용도에 맞게 잘 정렬해주면 될 것 같습니다. 정렬할 때는 보통 1) 중요한 컬럼을 좌측으로, 상대적으로 중요하지 않은 컬럼을 우측으로 옮겨주고, 2) 비슷한 역할을 하는 컬럼이나 서로간에 연관되어있는 컬럼은 같은 위치로 묶어줍니다. 최종적으로는 다음의 결과가 나와야 합니다.

ersion	event_category	view_category	view_id	view_action	in_app_event_category	in_app_event_label
4.0.4	goal	home	home	view	home.view	NaN
3.3.5	goal	gig	gig_detail	view	gig_detail.view	41201.0
4.0.4	goal	inbox	inbox_detail	view	inbox_detail.view	NaN
4.0.4	foreground	NaN	NaN	NaN	NaN	NaN
4.0.4	goal	buyer	buyer_order_track	view	buyer_order_track.view	NaN

In [882]:

```
# Write your code here!
# 위에서 명시한 형식에 맞게 컬럼의 순서를 변경합니다.
# 이를 oredered_columns라는 이름의 변수에 파이썬 리스트로 집어넣습니다.
oredered_columns = ['row_uuid', 'app_package_name', 'user_id',
                    'event_datetime', 'event_datetime_year', 'event_datetime_month', 'event_datetime
                    'event_datetime_hour', 'event_datetime_minute', 'event_datetime_second',
                    'device_manufacturer', 'device_type', 'os_type', 'os_version', 'app_version',
                    'event_category', 'view_category', 'view_id', 'view_action',
                    'in_app_event_category', 'in_app_event_label',
                    'source_type', 'channel', 'params_campaign', 'params_medium', 'params_term',
                    'is_first_activity', 'is_first_goal_activity']

raw_log = raw_log.reindex(columns=oredered_columns)
raw_log
```

Out[882]:

	row_uuid	app_package_name	user_id	event_datetime	event_datetime_y
0	fd2a188c-bc9b-4702-9c47-b546b2614817	com.kmong.iOS	F36FAA62-ADAC-4AA5-9B00-1FD6CB7EE957	2018-09-28 00:00:00+09:00	2
1	e62dccef-dd70-4415-8a33-c8324ddaed38	com.kmong.kmong	8a871e50-0717-4aed-9bad-04ac3c3793be	2018-09-28 00:00:00+09:00	2
2	14eb3197-db83-493a-b7be-83582960c40b	com.kmong.iOS	A9E5778A-8F3D-4597-9718-74BF953A9F64	2018-09-28 00:00:00+09:00	2
3	f9bb91af-248b-44dc-9f5c-1c00b37ea97b	com.kmong.iOS	168761CB-CB67-4592-867D-52780D651297	2018-09-28 00:00:01+09:00	2
4	236e9946-7801-4898-b609-06c8ab1139dc	com.kmong.iOS	ACABB7C0-4C76-413A-B314-E5D6DA0D0E5D	2018-09-28 00:00:02+09:00	2
...	...	...	...	...	
434239	9d9fd3d8-30c5-49e5-8480-41b536eb4edb	com.kmong.kmong	1fbc9ad2-8109-4294-ae9e-9acf6dce72c3	2018-09-29 23:59:52+09:00	2
434240	88653965-4aff-4d2e-8dd6-88f25ac82595	com.kmong.kmong	ea69fbd8-16fe-4db4-ae1e-512d5bbd1d6f	2018-09-29 23:59:53+09:00	2
434241	40bdc87e-265a-4cc2-a1d4-f1da96a0b9dc	com.kmong.kmong	1fbc9ad2-8109-4294-ae9e-9acf6dce72c3	2018-09-29 23:59:53+09:00	2
434242	f44a488b-1a8b-4f53-ad8e-8b21c695925b	com.kmong.kmong	ea69fbd8-16fe-4db4-ae1e-512d5bbd1d6f	2018-09-29 23:59:54+09:00	2

	row_uuid	app_package_name	user_id	event_datetime	event_datetime_y
434243	5fbb87c9-4d62-4eea-a75c-cab1a3af1d52	com.kmong.kmong	965b53e9-f6d6-4a9d-827e-e60c5eb52bb8	2018-09-29 23:59:58+09:00	2

434244 rows × 28 columns

10. funnel 데이터를 정리해주세요.

다음으로는 퍼널(funnel) 데이터를 정리하겠습니다. 퍼널 데이터에는 크몽 서비스에서 사용하는 퍼널에 대한 부연 설명을 담고 있습니다. 기쁘게도 이 데이터는 잘 정리되어 있기 때문에, 별도로 처리해줘야 하는 건 많지 않습니다. 다음의 설명을 참고하여 데이터를 정리해주세요.

1. 먼저 컬럼을 `Lv1`, `Lv2`, `viewid`, `viewid desc`, `funnel name`, `funnel desc` 순으로 정렬해주세요.
2. 그 다음 모든 컬럼을 소문자로 통일하고(ex: Lv1 -> lv1), 띄어쓰기를 언더바(_)로 바꿔주세요. (ex: funnel name -> funnel_name)

최종적으로는 다음의 결과가 나와야 합니다.

	lv1	lv2	view_id	view_desc	funnel_name	funnel_desc
0	11	1100	home	홈 (탭)	home	홈
1	12	1210	category_list	카테고리 목록 (탭)	category	카테고리
2	12	1200	category_gig	카테고리-상품목록	category	카테고리
3	13	1300	search	검색	search	검색
4	13	1301	search_gig	검색-상품목록	search	검색

In [883]:

```
# Write your code here!
raw_funnel = raw_funnel.reindex(columns=['Lv1', 'Lv2', 'viewid', 'viewid desc', 'funnel name', 'funr
raw_funnel.rename(columns={'Lv1':'lv1', 'Lv2':'lv2', 'viewid':'view_id', 'viewid desc':'view_desc',
raw_funnel
```

Out[883]:

	lv1	lv2	view_id	view_desc	funnel_name	funnel_desc
0	11	1100	home	홈 (탭)	home	홈
1	12	1210	category_list	카테고리 목록 (탭)	category	카테고리
2	12	1200	category_gig	카테고리-상품 목록	category	카테고리
3	13	1300	search	검색	search	검색
4	13	1301	search_gig	검색-상품목록	search	검색
5	13	1302	search_seller	검색-전문가	search	검색
6	14	1400	gig_detail	상품상세	gig	상품
7	14	1401	gig_detail_option	상품상세-상품 선택	gig	상품
8	14	1420	profile	전문가프로필	gig	상품
9	15	1520	login_sns	간편로그인	login	로그인
10	15	1500	login_email	간편로그인-이메일로그인	login	로그인
11	15	1510	find_account	간편로그인-이메일로그인-아이디/비밀번호-찾기	login	로그인
12	16	1600	inbox	메시지목록 (탭)	inbox	메시지
13	16	1610	inbox_detail	메시지목록-상세	inbox	메시지
14	16	1611	inbox_detail_orderRequest	메시지목록-상세-결제요청	inbox	메시지
15	17	1700	bookmarks_gig	메뉴목록-찜목록-서비스	bookmarks	찜하기
16	17	1701	bookmarks_portfolio	메뉴목록-찜목록-포트폴리오	bookmarks	찜하기
17	18	1801	order_addOption	상품상세-주문하기	order	결제
18	18	1800	order_payment	상품상세-주문하기-결제하기	order	결제
19	18	1810	kmongCash_charge	메뉴목록-크몽캐시-캐시충전하기	order	결제
20	19	1900	thankyou	결제완료	thankyou	결제완료

	lv1	lv2	view_id	view_desc	funnel_name	funnel_desc
21	20	2000	signup	개인/기업-회원가입	new	신규등록
22	20	2001	mobileAuth	휴대폰인증	new	신규등록
23	20	2020	seller_gig_register	메뉴목록-서비스등록	new	신규등록
24	22	2200	account_setting	메뉴목록-계정설정	account	계정관리 - 회원공통
25	22	2210	account_setting_notification	메뉴목록-계정설정-알림설정	account	계정관리 - 회원공통
26	22	2220	account_setting_AuthStatus	메뉴목록-계정설정-인증정보	account	계정관리 - 회원공통
27	22	2230	account_setting_pwd	메뉴목록-계정설정-비밀번호변경	account	계정관리 - 회원공통
28	22	2240	account_setting_profile	메뉴목록-계정설정-프로필	account	계정관리 - 회원공통
29	23	2320	seller_gig	메뉴목록-나의서비스	account_seller	계정관리 - 판매자
30	24	2400	menu	메뉴목록 (탭)	transaction_history	거래관리
31	24	2410	buyer_order_track	메뉴목록-구매관리-거래메시지	transaction_history	거래관리
32	24	2411	buyer_order_track_cancel	메뉴목록-구매관리-거래메시지-취소/문제해결	transaction_history	거래관리
33	24	2412	buyer_order_track_cancel_submit	메뉴목록-구매관리-거래메시지-취소/문제해결-제출	transaction_history	거래관리
34	24	2420	buyer_order_history	메뉴목록-구매관리	transaction_history	거래관리
35	24	2421	buyer_coupon_history	메뉴목록-쿠폰내역	transaction_history	거래관리
36	24	2430	seller_selling_history	메뉴목록-판매관리	transaction_history	거래관리
37	24	2431	seller_mileage_history	메뉴목록-마일리지 내역	transaction_history	거래관리
38	24	2432	seller_profits_history	메뉴목록-수익관리	transaction_history	거래관리
39	24	2450	buyer_payment_history	메뉴목록-결제내역	transaction_history	거래관리
40	24	2451	kmongCash_history	메뉴목록-크몽캐시	transaction_history	거래관리
41	24	2452	kmongCash_history_refundRequest	메뉴목록-크몽캐시-캐시환불	transaction_history	거래관리

	lv1	lv2	view_id	view_desc	funnel_name	funnel_desc
42	25	2510	about_howToUse	메뉴목록-크몽 이용방법	info	소개 및 안내
43	25	2520	termsOfService	메인하단-이용약관	info	소개 및 안내
44	25	2521	privacyPolicy	메인하단-개인정보처리방침	info	소개 및 안내
45	25	2530	tutorial	튜토리얼	info	소개 및 안내
46	27	2700	referrerProgram	메뉴목록-친구초대	referral_program	친구초대프로그램
47	28	2800	event	이벤트	event	이벤트프로모션
48	30	3010	kmongNotification	메뉴목록-크몽 안내	notification	알림
49	30	3011	kmongNotification_detail	메뉴목록-크몽 안내-상세	notification	알림
50	30	3000	notification	알림-알림	notification	알림
51	30	3001	notification_detail	알림-알림-상세	notification	알림
52	99	9910	NPS	NPS	others	기타

11. category 데이터를 정리해주세요.

퍼널(funnel) 데이터와 마찬가지로, category 데이터도 정리하겠습니다. category 데이터는크몽에서 판매하는 상품의 카테고리를 나타내는 정보인데, 퍼널 데이터와 마찬가지로 이 데이터로 잘 정리되어 있기 때문에 크게 건드려줘야 하는 부분은 없고, 컬럼명만 바꿔주면 될 것 같습니다. 다음의 기준을 참고하여 컬럼명을 사정해주세요.

- 단어 사이에는 언더바(_)를 붙입니다. (ex: categoryid -> category_id)
- 축약어는 전체 단어로 풀어서 작성합니다. (ex: cat1 -> category1)

최종적으로는 다음의 결과가 나와야 합니다.

	depth	category_id	category_name	category1_id	category2_id	category3_id	category1	category2	category
0	1	1	디자인	1	NaN	NaN	디자인	NaN	Nal
1	1	2	마케팅	2	NaN	NaN	마케팅	NaN	Nal
2	1	3	번역·통역	3	NaN	NaN	번역·통역	NaN	Nal
3	1	4	문서작성	4	NaN	NaN	문서작성	NaN	Nal
4	1	6	IT·프로그래밍	6	NaN	NaN	IT·프로그래밍	NaN	Nal

In [884]:

```
# Write your code here!
raw_category.rename(columns={'categoryid':'category_id', 'Lv2':'lv2', 'categoryname':'category_name'
raw_category
```

Out[884]:

	depth	category_id	category_name	category1_id	category2_id	category3_id	category1
0	1	1	디자인	1	NaN	NaN	디자인
1	1	2	마케팅	2	NaN	NaN	마케팅
2	1	3	번역·통역	3	NaN	NaN	번역·통역
3	1	4	문서작성	4	NaN	NaN	문서작성
4	1	6	IT·프로그래밍	6	NaN	NaN	IT·프로그래밍
...	...	...	...	...	...	...	...
240	3	72501	모델	7	725.0	72501.0	콘텐츠 제작
241	3	72502	MC	7	725.0	72502.0	콘텐츠 제작
242	3	72503	배우	7	725.0	72503.0	콘텐츠 제작
243	3	72504	공연	7	725.0	72504.0	콘텐츠 제작
244	3	72505	기타	7	725.0	72505.0	콘텐츠 제작

245 rows × 9 columns

세 개의 데이터를 하나로 합치기

이번에는 수정한 `log`, `funnel`, `category` 데이터를 하나로 합쳐보겠습니다. 이 부분은 다소 난이도가 높기 때문에, 적극적으로 구글을 검색하거나 다른 자료를 참고하는 것을 추천드립니다. 특히나 Merge, join, and concatenate (https://pandas.pydata.org/pandas-docs/stable/merging.html) 자료를 한 번 살펴보신 뒤 아래 문제를 시도한다면 더 쉽게 문제를 풀 수 있을 것입니다.

12. `log` 데이터의 view_id 컬럼과 `funnel` 데이터의 view_id 컬럼을 활용하여 두 개의 데이터를 합쳐주세요

앞서 설명드린대로 `log` 데이터에 사용자의 액티비티가 모두 모여져있고, 여기에서 가장 중요한 정보 중 하나인 퍼널(funnel)에 대한 부연 설명이 `funnel` 데이터에 모여 있습니다. 그러므로 이 두개를 합친다면, 차후에 `log` 데이터에서 퍼널 분석을 할 때 다른 데이터 분석가들이 더 직관적으로 데이터를 이해할 수 있을 것 같습니다.

그러므로 두 개의 데이터를 합쳐주세요. 두 개의 데이터와 연관이 있는 컬럼은 `view_id` 라는 이름의 컬럼이며, 데이터를 합칠 때는 merge (https://pandas.pydata.org/pandas-docs/stable/generated/pandas.DataFrame.merge.html)라는 함수를 잘 사용하면 생각보다 쉽게 데이터를 합칠 수 있습니다. 또한 합칠 때 `log` 데이터는 누락되지 않고 전부 살아있어야 하며, `funnel` 데이터는 `log` 데이터에 일치하는 것만 살리고 나머지는 지워져도 상관 없습니다. (이를 전문 용어로 **left merge**라고 합니다)

최종적으로는 다음의 결과가 나와야 합니다. (log 데이터와 동일해보이지만 자세히 보시면 오른쪽에 funnel 데이터가 추가되어 있습니다)

	row_uuid	app_package_name	user_id	event_datetime	event_datetime_year	event_datetime_mon
0	fd2a188c-bc9b-4702-9c47-b546b2614817	com.kmong.iOS	F36FAA62-ADAC-4AA5-9B00-1FD6CB7EE957	2018-09-27 15:00:00	2018	
1	e62dccef-dd70-4415-8a33-c8324ddaed38	com.kmong.kmong	8a871e50-0717-4aed-9bad-04ac3c3793be	2018-09-27 15:00:00	2018	
2	14eb3197-db83-493a-b7be-83582960c40b	com.kmong.iOS	A9E5778A-8F3D-4597-9718-74BF953A9F64	2018-09-27 15:00:00	2018	
3	f9bb91af-248b-44dc-9f5c-1c00b37ea97b	com.kmong.iOS	168761CB-CB67-4592-867D-52780D651297	2018-09-27 15:00:01	2018	
4	236e9946-7801-4898-b609-06c8ab1139dc	com.kmong.iOS	ACABB7C0-4C76-413A-B314-E5D6DA0D0E5D	2018-09-27 15:00:02	2018	

In [885]:

```
# Write your code here!
# left merge
merge1 = pd.merge(raw_log, raw_funnel, left_on='view_id', right_on='view_id', how='left')
merge1
```

Out[885]:

	row_uuid	app_package_name	user_id	event_datetime	event_datetime_yea
0	fd2a188c-bc9b-4702-9c47-b546b2614817	com.kmong.iOS	F36FAA62-ADAC-4AA5-9B00-1FD6CB7EE957	2018-09-28 00:00:00+09:00	201
1	e62dccef-dd70-4415-8a33-c8324ddaed38	com.kmong.kmong	8a871e50-0717-4aed-9bad-04ac3c3793be	2018-09-28 00:00:00+09:00	201
2	14eb3197-db83-493a-b7be-83582960c40b	com.kmong.iOS	A9E5778A-8F3D-4597-9718-74BF953A9F64	2018-09-28 00:00:00+09:00	201
3	f9bb91af-248b-44dc-9f5c-1c00b37ea97b	com.kmong.iOS	168761CB-CB67-4592-867D-52780D651297	2018-09-28 00:00:01+09:00	201
4	236e9946-7801-4898-b609-06c8ab1139dc	com.kmong.iOS	ACABB7C0-4C76-413A-B314-E5D6DA0D0E5D	2018-09-28 00:00:02+09:00	201
...	...	...	...	...	.
434239	9d9fd3d8-30c5-49e5-8480-41b536eb4edb	com.kmong.kmong	1fbc9ad2-8109-4294-ae9e-9acf6dce72c3	2018-09-29 23:59:52+09:00	201
434240	88653965-4aff-4d2e-8dd6-88f25ac82595	com.kmong.kmong	ea69fbd8-16fe-4db4-ae1e-512d5bbd1d6f	2018-09-29 23:59:53+09:00	201
434241	40bdc87e-265a-4cc2-a1d4-f1da96a0b9dc	com.kmong.kmong	1fbc9ad2-8109-4294-ae9e-9acf6dce72c3	2018-09-29 23:59:53+09:00	201
434242	f44a488b-1a8b-4f53-ad8e-8b21c695925b	com.kmong.kmong	ea69fbd8-16fe-4db4-ae1e-512d5bbd1d6f	2018-09-29 23:59:54+09:00	201
434243	5fbb87c9-4d62-4eea-a75c-cab1a3af1d52	com.kmong.kmong	965b53e9-f6d6-4a9d-827e-e60c5eb52bb8	2018-09-29 23:59:58+09:00	201

434244 rows × 33 columns

13. 12번 데이터의 in_app_event_label 컬럼과 category 데이터의 category_id 컬럼을 활용하여 두

개의 데이터를 합쳐주세요.

퍼널(funnel)과 마찬가지로 `category` 데이터도 하나로 합쳐보겠습니다.

이것도 12번 데이터와 마찬가지로 판다스(Pandas)의 merge (https://pandas.pydata.org/pandas-docs/stable/generated/pandas.DataFrame.merge.html), 그것도 **left merge**를 사용하면 됩니다. 다만 두 데이터가 `view_id` 라는 공통의 컬럼을 보유하고 있던 이전과는 다르게, 이번에는 12번 데이터에는 `in_app_event_label` 컬럼이, `category` 데이터에는 `category_id` 컬럼이 서로 연관이 있습니다. 이 컬럼을 사용하여 두 개를 합쳐주세요.

최종적으로는 다음의 결과가 나와야 합니다. (자세히 보시면 오른쪽에 `category_id` 데이터가 추가되어 있습니다)

e	funnel_desc	depth	category_id	category_name	category1_id	category2_id	category3_id	category1	category
e	홈	NaN	NaN	NaN	NaN	NaN	NaN	NaN	Na
g	상품	3.0	41201.0	자기소개서	4.0	412.0	41201.0	문서작성	자기소 서·이력
x	메시지	NaN	NaN	NaN	NaN	NaN	NaN	NaN	Na
N	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	Na
y	거래관리	NaN	NaN	NaN	NaN	NaN	NaN	NaN	Na

In [886]:

```
# Write your code here!
merge2 = pd.merge(merge1, raw_category, left_on='in_app_event_label', right_on='category_id', how='
merge2
```

Out[886]:

	row_uuid	app_package_name	user_id	event_datetime	event_datetime_yea
0	fd2a188c-bc9b-4702-9c47-b546b2614817	com.kmong.iOS	F36FAA62-ADAC-4AA5-9B00-1FD6CB7EE957	2018-09-28 00:00:00+09:00	201
1	e62dccef-dd70-4415-8a33-c8324ddaed38	com.kmong.kmong	8a871e50-0717-4aed-9bad-04ac3c3793be	2018-09-28 00:00:00+09:00	201
2	14eb3197-db83-493a-b7be-83582960c40b	com.kmong.iOS	A9E5778A-8F3D-4597-9718-74BF953A9F64	2018-09-28 00:00:00+09:00	201
3	f9bb91af-248b-44dc-9f5c-1c00b37ea97b	com.kmong.iOS	168761CB-CB67-4592-867D-52780D651297	2018-09-28 00:00:01+09:00	201
4	236e9946-7801-4898-b609-06c8ab1139dc	com.kmong.iOS	ACABB7C0-4C76-413A-B314-E5D6DA0D0E5D	2018-09-28 00:00:02+09:00	201
...	...	...	...	...	.
434239	9d9fd3d8-30c5-49e5-8480-41b536eb4edb	com.kmong.kmong	1fbc9ad2-8109-4294-ae9e-9acf6dce72c3	2018-09-29 23:59:52+09:00	201
434240	88653965-4aff-4d2e-8dd6-88f25ac82595	com.kmong.kmong	ea69fbd8-16fe-4db4-ae1e-512d5bbd1d6f	2018-09-29 23:59:53+09:00	201
434241	40bdc87e-265a-4cc2-a1d4-f1da96a0b9dc	com.kmong.kmong	1fbc9ad2-8109-4294-ae9e-9acf6dce72c3	2018-09-29 23:59:53+09:00	201
434242	f44a488b-1a8b-4f53-ad8e-8b21c695925b	com.kmong.kmong	ea69fbd8-16fe-4db4-ae1e-512d5bbd1d6f	2018-09-29 23:59:54+09:00	201
434243	5fbb87c9-4d62-4eea-a75c-cab1a3af1d52	com.kmong.kmong	965b53e9-f6d6-4a9d-827e-e60c5eb52bb8	2018-09-29 23:59:58+09:00	201

434244 rows × 42 columns

14. 합친 데이터에서 필요하지 않은 컬럼은 버려주세요.

이제 데이터를 다 정리하고 합쳤으면, 나머지는 분석에 필요하지 않은 데이터를 버려주는 일만 남았습니다. 실제 회사에서는 데이터를 처음 읽을 때 굉장히 많은 컬럼을 읽어오게 되는데, 이를 전부 읽어와 분석하지 않고 1) 필요하지 않은 컬럼은 사전에 버려주고, 2) 남은 데이터에서 분석할 때 필요한 데이터만 그때그때 가져오는 습관을 드리는 것이 좋습니다.

그러므로 다음의 컬럼은 사용하지 않을 예정이니 데이터에서 삭제하도록 하겠습니다.

- in_app_event_category
- in_app_event_label
- source_type
- Lv1, Lv2
- funnel_name, depth
- category_id, category1_id, category2_id, category3_id

최종적으로는 다음의 결과가 나와야 합니다.

event_datetime_year	event_datetime_month	event_datetime_day	event_datetime_hour	event_datetime_minute	ev
2018	9	27	15	0	
2018	9	27	15	0	
2018	9	27	15	0	
2018	9	27	15	0	
2018	9	27	15	0	

In [887]:

```
# Write your code here!

merge2 = merge2.drop(['in_app_event_category','in_app_event_label',
                      'source_type', 'lv1', 'lv2', 'funnel_name', 'depth', 'category_id',
                      'category1_id', 'category2_id', 'category3_id'
                     ], axis=1)
merge2
```

Out[887]:

	row_uuid	app_package_name	user_id	event_datetime	event_datetime_yea
0	fd2a188c-bc9b-4702-9c47-b546b2614817	com.kmong.iOS	F36FAA62-ADAC-4AA5-9B00-1FD6CB7EE957	2018-09-28 00:00:00+09:00	201
1	e62dccef-dd70-4415-8a33-c8324ddaed38	com.kmong.kmong	8a871e50-0717-4aed-9bad-04ac3c3793be	2018-09-28 00:00:00+09:00	201
2	14eb3197-db83-493a-b7be-83582960c40b	com.kmong.iOS	A9E5778A-8F3D-4597-9718-74BF953A9F64	2018-09-28 00:00:00+09:00	201
3	f9bb91af-248b-44dc-9f5c-1c00b37ea97b	com.kmong.iOS	168761CB-CB67-4592-867D-52780D651297	2018-09-28 00:00:01+09:00	201
4	236e9946-7801-4898-b609-06c8ab1139dc	com.kmong.iOS	ACABB7C0-4C76-413A-B314-E5D6DA0D0E5D	2018-09-28 00:00:02+09:00	201
...	...	...	...	...	.
434239	9d9fd3d8-30c5-49e5-8480-41b536eb4edb	com.kmong.kmong	1fbc9ad2-8109-4294-ae9e-9acf6dce72c3	2018-09-29 23:59:52+09:00	201
434240	88653965-4aff-4d2e-8dd6-88f25ac82595	com.kmong.kmong	ea69fbd8-16fe-4db4-ae1e-512d5bbd1d6f	2018-09-29 23:59:53+09:00	201
434241	40bdc87e-265a-4cc2-a1d4-f1da96a0b9dc	com.kmong.kmong	1fbc9ad2-8109-4294-ae9e-9acf6dce72c3	2018-09-29 23:59:53+09:00	201
434242	f44a488b-1a8b-4f53-ad8e-8b21c695925b	com.kmong.kmong	ea69fbd8-16fe-4db4-ae1e-512d5bbd1d6f	2018-09-29 23:59:54+09:00	201
434243	5fbb87c9-4d62-4eea-a75c-cab1a3af1d52	com.kmong.kmong	965b53e9-f6d6-4a9d-827e-e60c5eb52bb8	2018-09-29 23:59:58+09:00	201

434244 rows × 31 columns

15. `row_uuid` 를 인덱스(index)로 지정해주세요.

데이터를 다 정리했으면 마지막으로 인덱스(index)를 지정하고 마무리하겠습니다. 현재 가지고 있는 데이터에서 인덱스가 될 수 있을만한 컬럼은 `row_uuid` 입니다. 이 컬럼을 인덱스로 지정하면, 앞으로 데이터의 행(row)을 가져오고 분석할 때 큰 도움이 될 것 같습니다. 최종적으로는 다음의 결과가 나와야 합니다.

id	app_package_name	user_id	event_datetime	event_datetime_year	event_datetime_month	event_datetim
c- 2- 7- l7	com.kmong.iOS	F36FAA62-ADAC-4AA5-9B00-1FD6CB7EE957	2018-09-27 15:00:00	2018	9	
ɜf- 5- 3- 38	com.kmong.kmong	8a871e50-0717-4aed-9bad-04ac3c3793be	2018-09-27 15:00:00	2018	9	
7- a- e-)b	com.kmong.iOS	A9E5778A-8F3D-4597-9718-74BF953A9F64	2018-09-27 15:00:00	2018	9	
ıf- c- c- 'b	com.kmong.iOS	168761CB-CB67-4592-867D-52780D651297	2018-09-27 15:00:01	2018	9	
6- 8- 9- dc	com.kmong.iOS	ACABB7C0-4C76-413A-B314-E5D6DA0D0E5D	2018-09-27 15:00:02	2018	9	

In [888]:

```
# Write your code here!
merge2.set_index('row_uuid', inplace=True)
merge2
data=merge2
data
```

Out[888]:

row_uuid	app_package_name	user_id	event_datetime	event_datetime_year	event_
fd2a188c-bc9b-4702-9c47-b546b2614817	com.kmong.iOS	F36FAA62-ADAC-4AA5-9B00-1FD6CB7EE957	2018-09-28 00:00:00+09:00	2018	
e62dccef-dd70-4415-8a33-c8324ddaed38	com.kmong.kmong	8a871e50-0717-4aed-9bad-04ac3c3793be	2018-09-28 00:00:00+09:00	2018	
14eb3197-db83-493a-b7be-83582960c40b	com.kmong.iOS	A9E5778A-8F3D-4597-9718-74BF953A9F64	2018-09-28 00:00:00+09:00	2018	
f9bb91af-248b-44dc-9f5c-1c00b37ea97b	com.kmong.iOS	168761CB-CB67-4592-867D-52780D651297	2018-09-28 00:00:01+09:00	2018	
236e9946-7801-4898-b609-06c8ab1139dc	com.kmong.iOS	ACABB7C0-4C76-413A-B314-E5D6DA0D0E5D	2018-09-28 00:00:02+09:00	2018	
...	...	...	...	...	
9d9fd3d8-30c5-49e5-8480-41b536eb4edb	com.kmong.kmong	1fbc9ad2-8109-4294-ae9e-9acf6dce72c3	2018-09-29 23:59:52+09:00	2018	
88653965-4aff-4d2e-8dd6-88f25ac82595	com.kmong.kmong	ea69fbd8-16fe-4db4-ae1e-512d5bbd1d6f	2018-09-29 23:59:53+09:00	2018	
40bdc87e-265a-4cc2-a1d4-f1da96a0b9dc	com.kmong.kmong	1fbc9ad2-8109-4294-ae9e-9acf6dce72c3	2018-09-29 23:59:53+09:00	2018	
f44a488b-1a8b-4f53-ad8e-8b21c695925b	com.kmong.kmong	ea69fbd8-16fe-4db4-ae1e-512d5bbd1d6f	2018-09-29 23:59:54+09:00	2018	
5fbb87c9-4d62-4eea-a75c-cab1a3af1d52	com.kmong.kmong	965b53e9-f6d6-4a9d-827e-e60c5eb52bb8	2018-09-29 23:59:58+09:00	2018	

434244 rows × 30 columns

16. data 변수에 할당된 데이터의 행렬 사이즈를 출력하세요 .

print(data.shape) 한 결과

In [890]:

```
print(data.shape)
```

```
(434244, 30)
```

[제출]

1. 위 마지막 작업 결과를 csv 파일로 저장합니다.
2. 현재 파일을 pdf 파일을 만들어 제출합니다.