2015 년 서울시 축제 분석

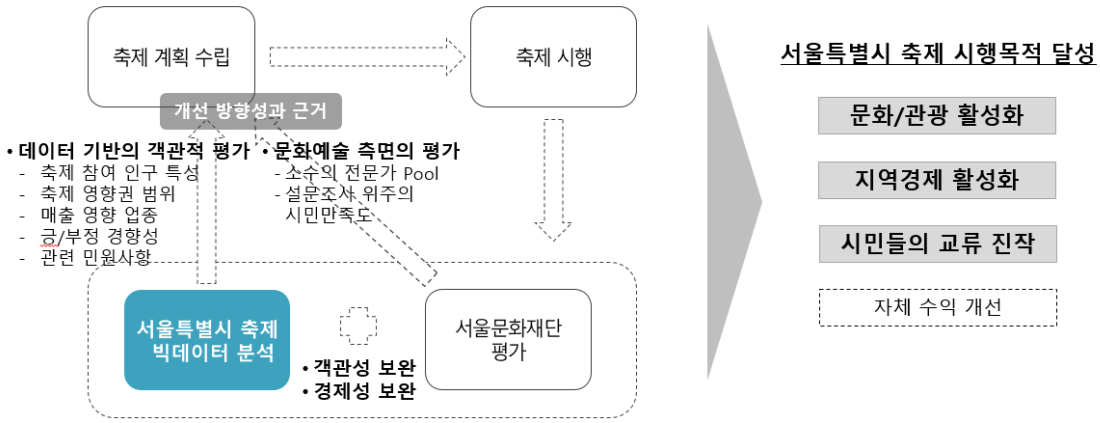
서울시 빅데이터 캠퍼스

서울시에서는 2015 년 서울형 빅데이터 공유/활용 플랫폼 구축의 일환으로 문화 예술 가치 위주의 기존 축제 평가를 객관성과 경제성 측면에서 보완하고, 정량적인 분석을 통해 차년도 축제 계획 수립시에 개선 방향성에 대한 지침으로 삼고자 축제 빅데이터 분석을 시행하였습니다. 2015 년 당시에는 엑셀을 사용하여 축제 전년도와 축제년도간의 유동인구 증감 비교, 축제지역 주변 매출 증감 변화 등을 분석하였습니다. 이러한 분석 결과중 하둡 등 일반인들이 사용하기 어려운 소프트웨어를 사용하는 전처리 부분을 제외하고, 전처리 결과로 추출된 데이터를 활용하여 빅데이터 캠퍼스에서 활용 가능한 분석툴인 R 로 분석을 체험하시도록 도와 드림으로써, 여러분들이 빅데이터 캠퍼스 분석환경을 보다 쉽게 이용하고, 데이터를 활용하시는데 참고가 되시기를 바랍니다

2015 년 서울시 축제 분석 개요

2015 년 이전의 축제 평가 방식을 데이터에 근거한 객관적인 방식으로 전환이 가능한지를 검토함으로써, 효과 검증시, 이번 실증 차원의 분석결과를 확산함으로써, 향후의 축제계획 수립시에 근거로 활용하기 위한 것입니다.

이를 통해 문화/관광 활성화, 지역경제 활성화, 시민들의 교류 진작 및 축제 자체의 수익개선 등을 기대할 수 있으리라 생각됩니다.



분석 대상 축제 개요

분석 대상 축제는 2015 년도에 시행된 시민문화 교류형 축제 1 건, 전문 예술 문화형 축제 2 건, 관광마케팅산업형으로 분류되는 축제 2 건 등, 총 5 건이며, 각각의 축제 행사 명칭과 일정, 기간 등은 아래의 그램과 같습니다. 가장 긴 축제가 17 일간 열린 서울 빛초롱 축제이며, 이외의 축제는 모두 2~4 일의 기간이며, 모두 주말에 행사가 진행되었습니다.

행사명	축제평가 유형	행사월	행사일정	행사기간
· 서울 문화의 밤	시민문화교류형	8월	2015년08월28일(금)~2015년08월29(토)	2일
· 서울드럼페스티벌	전문예술문화형	8월	2015년08월07일(금)~2015년08월08(토)	2일
· 하이서울페스티벌	전문예술문화형	10월	2015년10월01일(목)~2015년10월04(일)	4일
· 서울빛초롱축제	관광마케팅산업형	11월	2015년11월06일(금)~2015년11월22(일)	17일
· 서울김장문화제	관광마케팅산업형	11월	2015년11월06일(금)~2015년11월08(일)	3일

활용 데이터

야외 축제에는 강수 등의 날씨가 도와주는 것이 행사 흥행의 가장 중요한 요소입니다. 2014 년 대비 2015 년 축제기간내 비오는 날이 2 배가 많아서 2015 년 일부 축제는 흥행이 낮아진 것으로 나타났습니다만, 이 분석에서는 그러한 내용보다는 KT 등의 통신사에서 생성하고, 서울시 빅데이터 캠퍼스에서 시민 여러분들에게 이용 편리를 제공해 드릴 수 있는 이동전화 통화로그에 기반하고 있는 서울시 유동인구 데이터와 신한카드 등의 신용카드사에서 생성하고, 역시 서울시 빅데이터 캠퍼스에서 사용하실 수 있는 신용카드 매출 데이터를 활용하여 분석하는 사례를 가지고 분석을 진행할 예정입니다.

사용하는 데이터의 주요 내역은 아래의 그림과 같습니다.

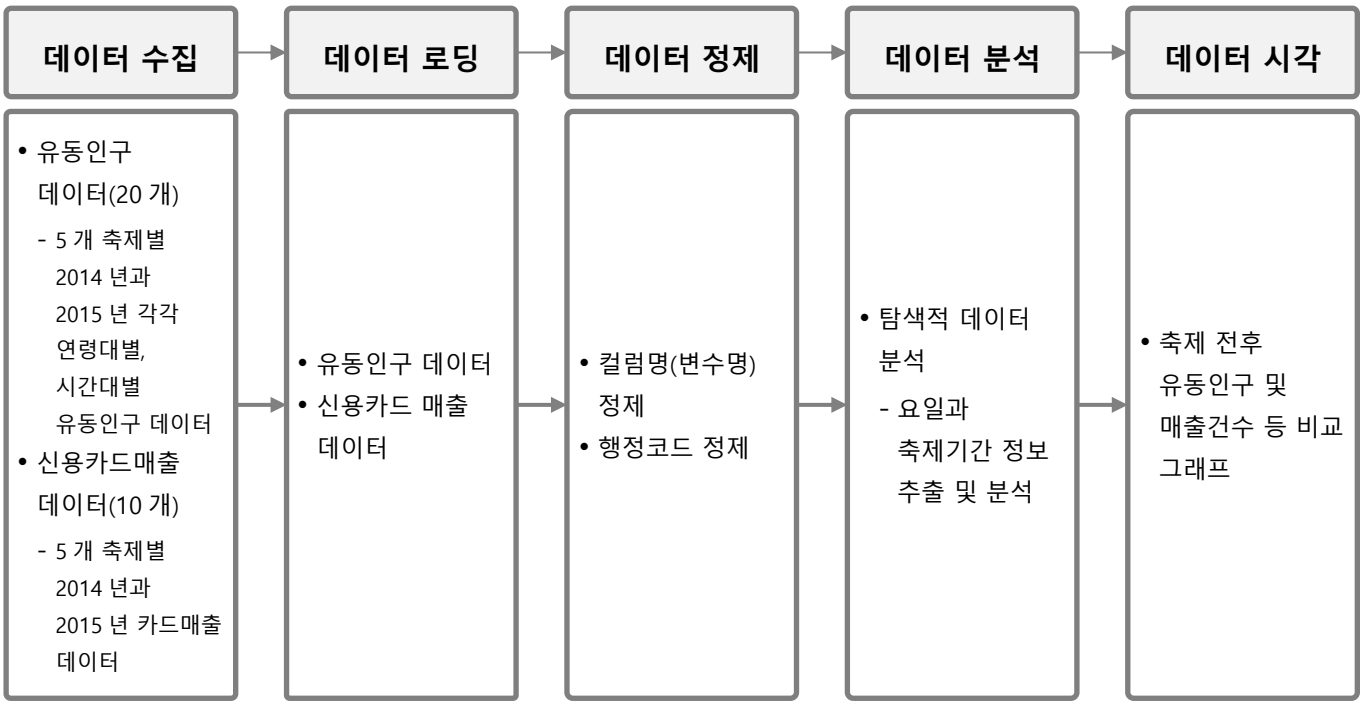
데이터 명	• 유동인구 데이터	원천기관	• 통신사 (SKT 등)
데이터 내역	• 블록코드, 일자, 시간대, 성, 연령, 유입지, 유동인구수		
활용 목적	• 서울특별시 축제의 경제적 평가 중 유입인구에 대해 평가 • 유입인구의 규모, 유입지, 특성 등을 향후 축제 콘텐츠 개선과 마케팅에 활용 • 서울문화재단의 기존 평가프로세스와 병합하여 개선		
데이터 명	• 매출 데이터	원천기관	• 신용카드 회사 (신한카드 등)
데이터 내역	• 블록코드, 일자, 시간대, 성, 연령, 유입지, 업종, 매출금액		
활용 목적	• 서울특별시 축제의 경제적 평가 중 인근 상권 매출에 대해 평가 • 매출 관련 데이터를 기반으로 인근 상인들의 민원 해소 • 업종별 매출 특성을 파악하여 향후 축제 콘텐츠 개선과 마케팅에 활용 • 서울문화재단의 기존 평가프로세스와 병합하여 개선		

분석 프로세스 및 적용 분석 툴

매월 수억통이상의 통화로그와 수천만건 이상이 생성되는 카드승인데이터에서 축제지역 인근에서 발생한 실적들만을 골라서 추출하기위해서는 대량의 데이터를 신속히 처리하는 기술의 적용이 필요하기 때문에 대용량 분산처리 환경인 하둡 기반으로 작업을 수행한 다음, 분석 데이터로 정제를 한 후, 2014 년과 2015 년 비교를 통해 효과를 분석하는 프로세스로 작업을 진행하였으며, 전체적인 분석 프로세스는 아래와 같습니다.

이번 분석에서는 전체 흐름중에서 축제 전후 유동인구 변화 및 축제 인근의 상가 혹은 점포의 매출 변화 위주로 진행합니다.

분석내용을 R 로 어떻게 구현하는지, 실제 데이터를 다뤄 보시면서 분석하는 즐거움을 느껴보실 수 있도록 준비하였습니다.



2015 년 시행 서울시 축제 효과 분석

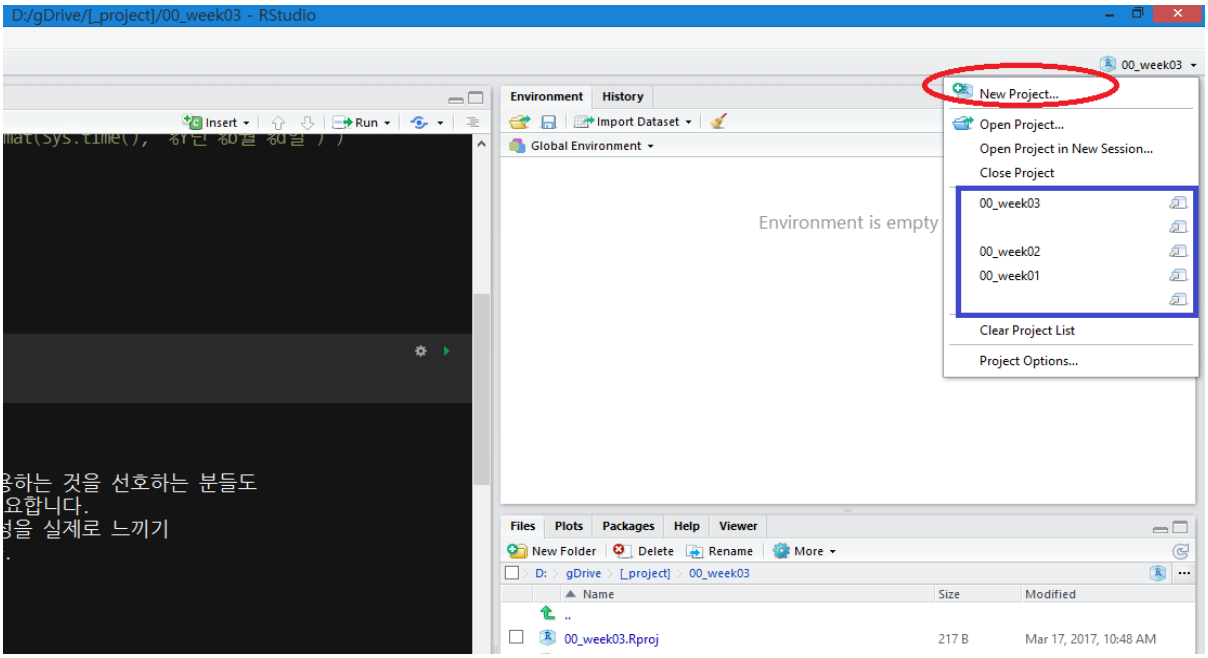
빅데이터 캠퍼스의 파일서버에서 FTP 로 해당 데이터를 다운로드 받아 R Studio 의 프로젝트 폴더에 직접 혹은 다운로드 폴더에서 R Studio 에서 접근이 가능한 작업영역(컴퓨터 용어로는 작업디렉토리라고 합니다.)으로 데이터를 복사해 놓아야 사용이 가능하게 됩니다.

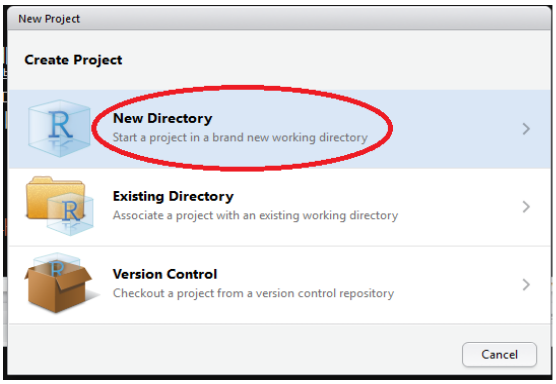
이 작업을 위해서는 먼저 R Studio 에서 이야기 하는 프로젝트가 뭔지, 어떻게 지정하고 사용하는지를 먼저 아셔야 합니다.

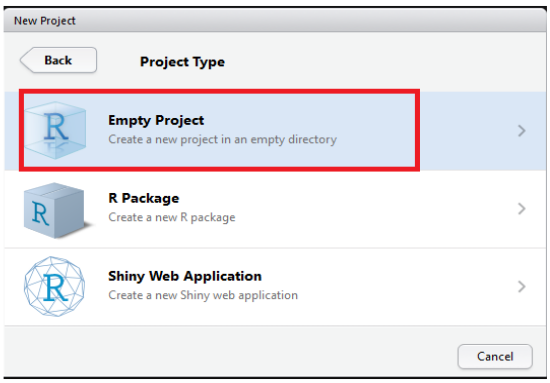
R Studio Project 관리

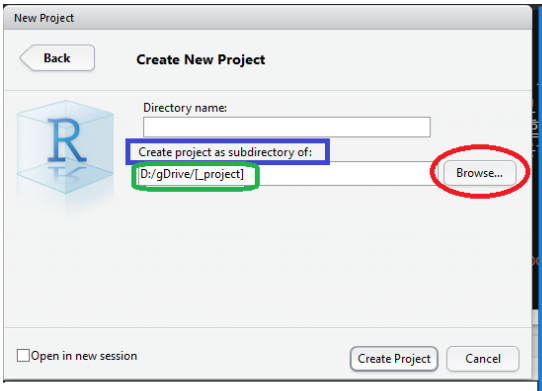
작업 디렉토리 관리는 아주 기초적인 것이라 엄청나게 중요하긴 합니다만, 그 중요성을 실제로 느끼기 어렵습니다. 그래서 추천드리는 방법은 R Studio 의 프로젝트 관리 기능을 활용하시는 것입니다.

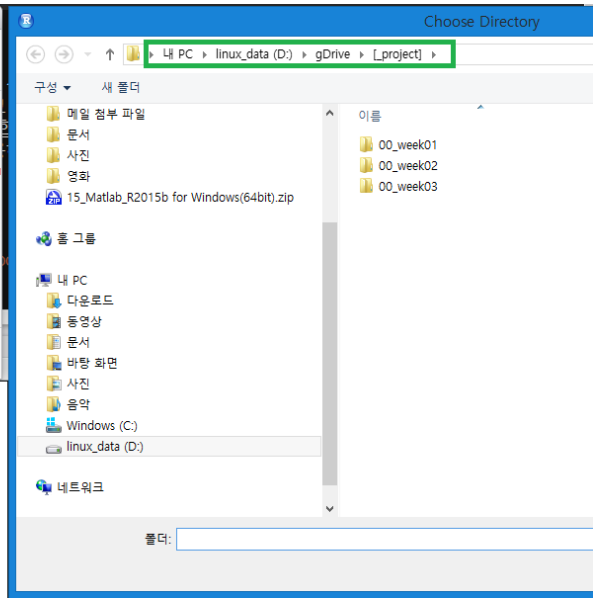
방법도 그렇게 어렵지는 않습니다. 아래 그림과 같이 윈도우즈의 탐색기 사용하시듯 하시면 됩니다.



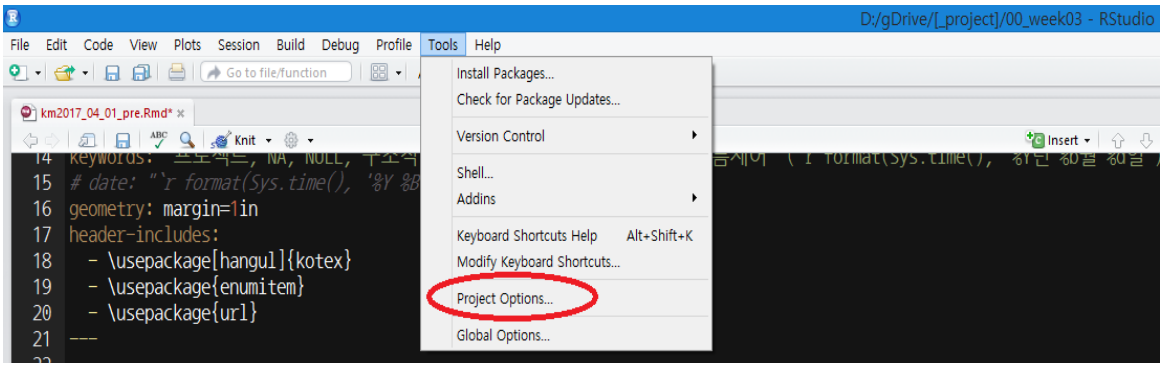


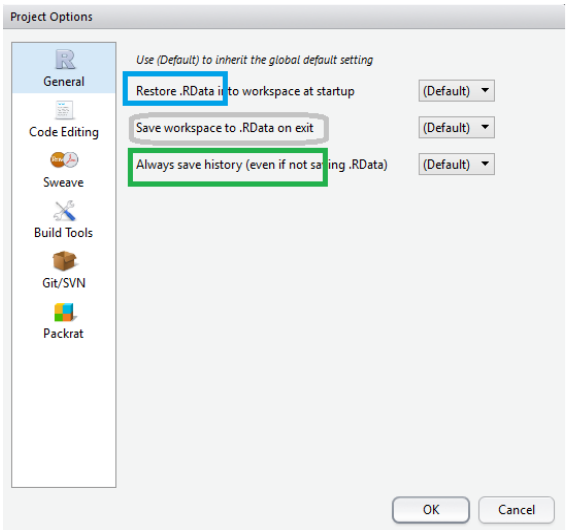


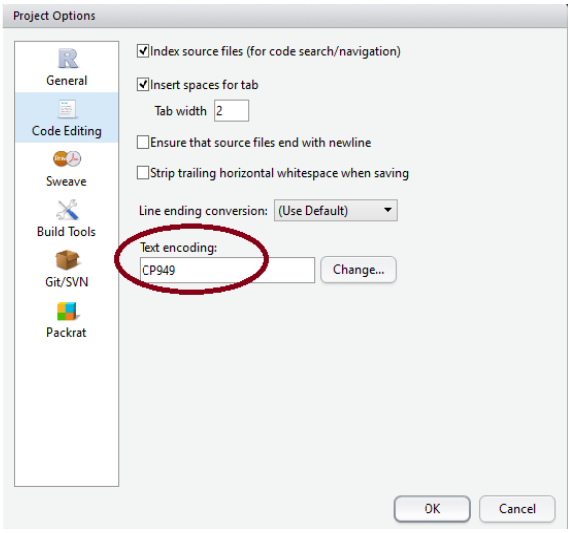




프로젝트는 실제로 업무 단위라고 보시면 되기 때문에, 업무별 성격에 따라 별도의 속성을 관리할 수 있도록 메인 메뉴에서 프로젝트 별로도 옵션을 설정할 수 있게 해 줍니다.



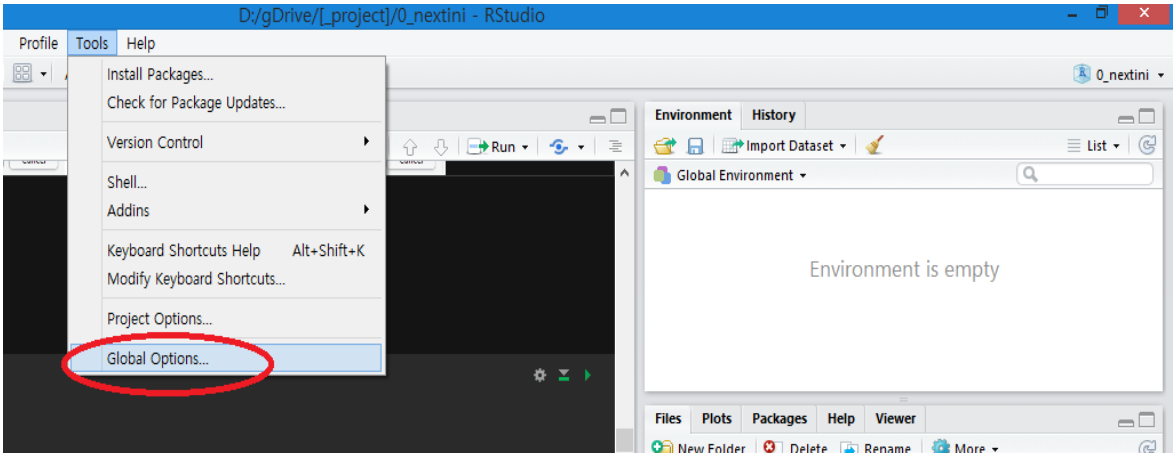


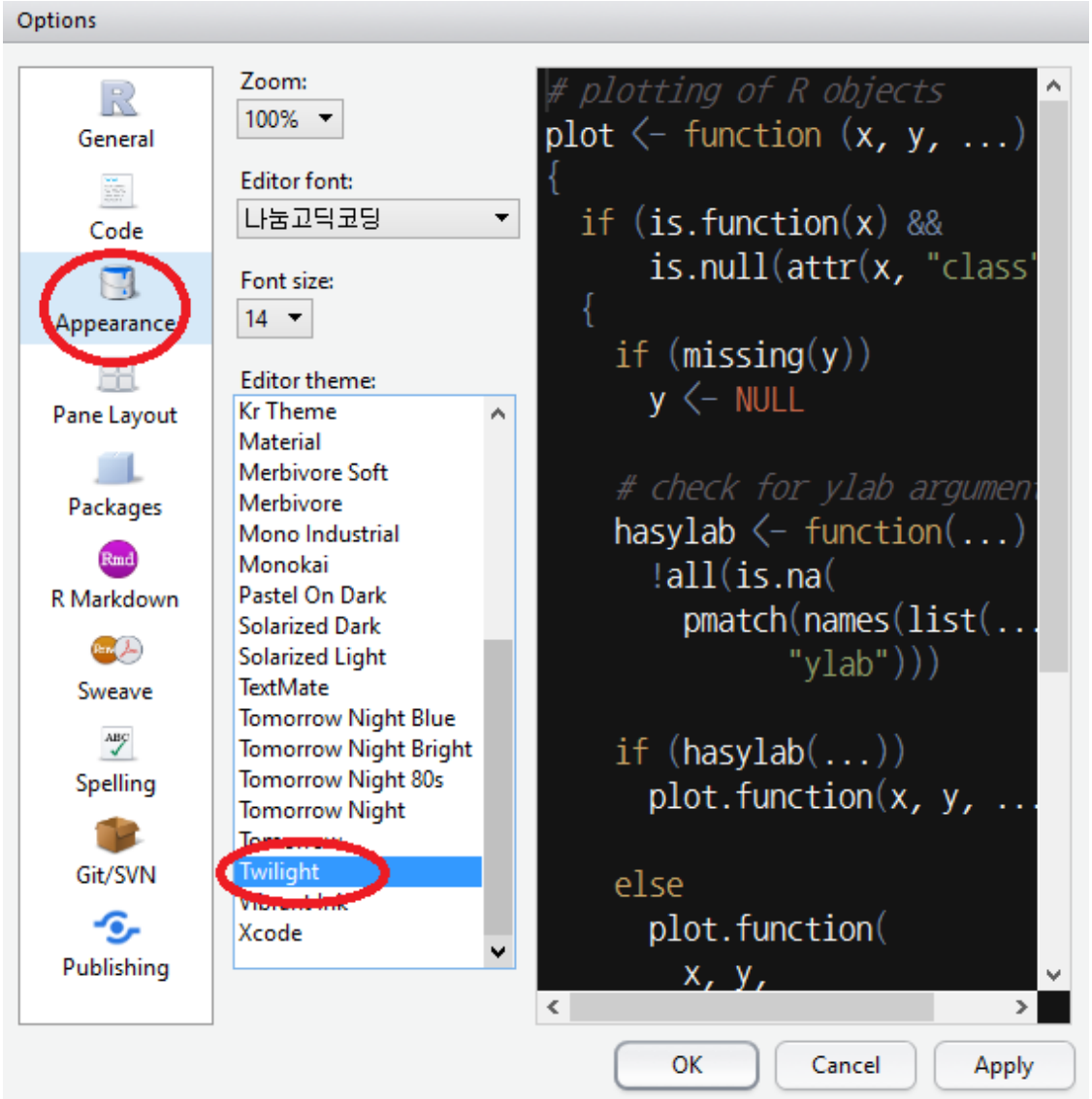


예시에서는 0_nextini 라는 프로젝트를 생성하였습니다. 여러분들은 본인의 분석을 위한 임의의 프로젝트를 만드시면 됩니다.

그럼 이제 분석을 위한 준비가 된 것입니다.

본격적인 분석에 앞서, 왜 여기 그림은 옆에 검정색 바탕에 흰색 글씨가 보이는데, 나는 왜 없지? 하시는 분이 계실 지도 몰라서 간단하게 코드 편집 화면을 변경하는 방법을 알려드리겠습니다. R Studio 상단메뉴의 Tools > Global Options 를 선택하신후 열리는 팝업창에서 좌측 Appearance 를 클릭하신 후, Font size 선택 톱다운 메뉴 하단에 있는 Editor theme 선택 메뉴에서 스크롤하셔서 "Twilight"를 선택하시면 됩니다.





데이터 로딩(Loading)

분석을 위한 가장 중요한 첫 단계는 데이터 확보입니다. 이렇게 확보된 데이터를 R Studio 에서 사용하기 위해서는 메모리상에 올려 데이터 객체(Object)로 만들어주어야 합니다. 이러한 단계를 보통 데이터 로딩(Loading)이라고 하는데, 보통 CSV(Comma Separated Value) 파일로 받아서, 외부 데이터를 읽어 들이는 R 의 함수를 사용하여 데이터를 로딩하는데, 그함수가 바로 read.csv() 함수입니다.

```
biz_group <- read.csv("biz_group.csv", stringsAsFactors = F)
```

그런 다음 보통은 dimension 함수로 데이터의 행의 객수나 열의 갯수를 확인한 후, 데이터가 길 경우에는 head() 함수로 제일 윗 부분 6 개 행의 모습을 확인하여, 데이터가 제대로 로딩 되었는지 등을 확인하는 것이 일반적입니다.

```
dim(biz_group)
```

```
## [1] 63  2
```

```
head(biz_group)
```

```
##             업종 업종군
## 1 입시보습학원    교육
## 2   외국어학원    교육
## 3   예체능학원    교육
## 4     슈퍼마켓 도소매
## 5       정육점 도소매
## 6       편의점 도소매
```

하지만 읽어 들여야 할 데이터가 많은 경우는 어떻게 해야할까요? 100 개면 100 번을 다 함수로 읽어 들여야 할까요? 약간의 R 프로그래밍 지식을 활용하시면 이런 난관을 극복하실 수 있습니다.

분석용 데이터 현황

현재 읽어 들여야 하는 데이터 수는 유동인구 관련 5 개 축제별 2014 년과 2015 년 각각 연령대별, 시간대별 유동인구 데이터가 도합 20 개이며, 신용카드 매출 관련 5 개 축제별 2014 년과 2015 년 데이터가 도합 10 개 입니다.

복사해서 붙여넣기를 30 번하는 것도 쉽지 않은 일이므로, R 의 기능을 활용하여 일괄적으로 데이터 객체로 읽어들이는 방법을 사용하는 것이 시간절약도 되고 편리하기도 합니다.

먼저 유동인구 데이터는 flowdata 서브디렉토리를 만들어 저장하고, 카드매출 데이터는 revenuedata 서브디렉토리에 각각 저장합니다.

read.csv() 함수의 첫번째 인자로 파일명을 집어넣어서 읽어 들이기 때문에 각 디렉토리에 저장된 데이터파일의 파일명을 읽어들여 변수에 저장해주는 dir() 함수를 사용하여 파일명을 읽어들여, 각각 flowdata 라는 변수와 revdata 라는 변수에 할당해줍니다.

아래 코드에서 서브디렉토리의 지정은 반드시 따옴표로 둘러싼 문자열로 지정을 해주어야 하며, 현재 작업디렉토리 하단에 있는 서브디렉토리라는 의미로 디렉토리명앞에 "./"를 반드시 붙여 주셔야 합니다.

즉, dir("./flowdata")는 현재 프로젝트 작업 디렉토리 밑에 있는 flowdata 디렉토리에 저장되어 있는 파일들의 파일명을 읽어들이라는 명령어입니다.

```
flowlist <- dir("./flowdata/")
flowlist
```

```
##  [1] "CULTURE_AGE_INFLOW_2014_NEW.txt"  "CULTURE_AGE_INFLOW_2015_NEW.txt"
##  [3] "CULTURE_TIME_INFLOW_2014_NEW.txt" "CULTURE_TIME_INFLOW_2015_NEW.txt"
##  [5] "DRUM_AGE_INFLOW_2014_NEW.txt"     "DRUM_AGE_INFLOW_2015_NEW.txt"
##  [7] "DRUM_TIME_INFLOW_2014_NEW.txt"    "DRUM_TIME_INFLOW_2015_NEW.txt"
##  [9] "HISEOUL_AGE_INFLOW_2014_NEW.txt"  "HISEOUL_AGE_INFLOW_2015_NEW.txt"
## [11] "HISEOUL_TIME_INFLOW_2014_NEW.txt" "HISEOUL_TIME_INFLOW_2015_NEW.txt"
## [13] "KIMJANG_AGE_INFLOW_2014_NEW.txt"  "KIMJANG_AGE_INFLOW_2015_NEW.txt"
## [15] "KIMJANG_TIME_INFLOW_2014_NEW.txt" "KIMJANG_TIME_INFLOW_2015_NEW.txt"
## [17] "LAMP_AGE_INFLOW_2014_NEW.txt"     "LAMP_AGE_INFLOW_2015_NEW.txt"
## [19] "LAMP_TIME_INFLOW_2014_NEW.txt"    "LAMP_TIME_INFLOW_2015_NEW.txt"
```

```
revlist <- dir("./revenuedata/")
revlist
```

```
##  [1] "CULTURE_CARD_2014_215474_NEW.txt" "CULTURE_CARD_2015_298037_NEW.txt"
##  [3] "DRUM_CARD_2014_106822_NEW.txt"    "DRUM_CARD_2015_135082_NEW.txt"
##  [5] "HISEOUL_CARD_2014_436877_NEW.txt" "HISEOUL_CARD_2015_517691_NEW.txt"
##  [7] "KIMJANG_CARD_2014_349363_NEW.txt" "KIMJANG_CARD_2015_431704_NEW.txt"
##  [9] "LAMP_CARD_2014_753748_NEW.txt"    "LAMP_CARD_2015_911871_NEW.txt"
```

위에서 보신 바와 같이 유동인구 관련 데이터 파일 20 개, 카드매출 관련 데이터 매출 10 개 등이 있습니다. 보통 이러한 경우에는 for 문이라는 흐름제어(flow control) 기법을 활용하여, 프로그래밍적으로 처리합니다.

유동인구 데이터 Loading

for 문은 주어진 데이터(R 에서는 이를 벡터라고 이야기 합니다.)를 첫번째 원소부터 마지막 원소까지 순차적으로 돌려가면서 실행해주는 함수로 흔히 루프(Loop)라고 표현하는 흐름제어(flow control) 구문입니다.

for 문을 활용하여 데이터를 읽어 들이기전에 한가지 미리 해두어야 하는 작업이 있습니다. 데이터를 읽어들여서 저장할 변수 이름을 미리 만들어 두는 것이 필요합니다. 그냥 f1, f2, f20 과 같이 만들어도 당장은 큰 문제가 안됩니다만, 나중에 다시 알아보기가 매우 어려울 수가 많습니다.

그럼 어떻게 하면 좋을까요? 이번 경우에는 변수명을 디렉토리에 들어 있는 파일명의 일부로 하여, 한 참 시간이 지난 후에도 변수명을 보고, 대략 어떤 축제 관련 몇 년도 데이터 인지 알 수 있도록 하는 것이 좋을 것 같습니다.

이와 같이 특정 문자열의 일부를 추출하고자 하는 문자열의 시작 위치 번호와 종료 위치번호를 지정하여 뽑아주는 함수가 substr() 함수입니다.

아래에서 nchar()함수는 문자열의 객수를 추출하는 함수로 nchar("CULTURE_AGE_INFLOW_2014_NEW.txt") 함수를 호출하면, 31 을 반환(return)해 줍니다. 즉, 글자의 갯수를 세어주는 기능을 합니다. 여기에 특정한 숫자를 빼주면 그 숫자만큼 문자열의 마지막에서 앞자리에 있는 문자의 위치를 나타냅니다.

따라서, 아래와 같이 8 을 빼주면 뒤에 있는 7 개 글자는 빼고, 그 앞에 있는 글자까지 추출하라는 의미가 됩니다.

따라서 아래 명령어 Expression 의 의미는 flowlist 에 저장된 파일명 원소 각각에서 끝부분 7 자리를 제외하고, 첫글자 부터 추출하라는 의미가 됩니다.

결과를 보시면 각 파일명에 끝 부분에 공통적으로 들어 있는 "_NEW.txt"를 제외한 나머지 문자열이 추출된 것을 확인하실 수 있습니다.

```
flow_nm <- substr(flowlist,1,nchar(flowlist)-8)
flow_nm
```

```
##  [1] "CULTURE_AGE_INFLOW_2014"  "CULTURE_AGE_INFLOW_2015"
##  [3] "CULTURE_TIME_INFLOW_2014" "CULTURE_TIME_INFLOW_2015"
##  [5] "DRUM_AGE_INFLOW_2014"     "DRUM_AGE_INFLOW_2015"
##  [7] "DRUM_TIME_INFLOW_2014"    "DRUM_TIME_INFLOW_2015"
##  [9] "HISEOUL_AGE_INFLOW_2014"  "HISEOUL_AGE_INFLOW_2015"
## [11] "HISEOUL_TIME_INFLOW_2014" "HISEOUL_TIME_INFLOW_2015"
## [13] "KIMJANG_AGE_INFLOW_2014"  "KIMJANG_AGE_INFLOW_2015"
## [15] "KIMJANG_TIME_INFLOW_2014" "KIMJANG_TIME_INFLOW_2015"
## [17] "LAMP_AGE_INFLOW_2014"     "LAMP_AGE_INFLOW_2015"
## [19] "LAMP_TIME_INFLOW_2014"    "LAMP_TIME_INFLOW_2015"
```

R 에서는 일렬로 순차적으로 정렬된 데이터를 벡터라고 한다고 위에서 간략히 말씀드렸는데, 이렇게 순서있게 정리된 데이터에서 원하는 위치에 있는 원소만 추출할 때는 변수명 옆에 대괄호를 치고, 해당위치에 숫자를 넣어주셔야 합니다.

아래에 보시는 바와 같이 전체 벡터 데이터 중에서 1 번째, 7 번째, 20 번째 원소의 내용을 filelist 와 flow_nm 벡터에서 추출할 수 있습니다.

```
flowlist[1]
```

```
## [1] "CULTURE_AGE_INFLOW_2014_NEW.txt"
```

```
flow_nm[1]
```

```
## [1] "CULTURE_AGE_INFLOW_2014"
```

```
flowlist[7]
```

```
## [1] "DRUM_TIME_INFLOW_2014_NEW.txt"
```

```
flow_nm[7]
```

```
## [1] "DRUM_TIME_INFLOW_2014"
```

```
flowlist[20]
```

```
## [1] "LAMP_TIME_INFLOW_2015_NEW.txt"
```

```
flow_nm[20]
```

```
## [1] "LAMP_TIME_INFLOW_2015"
```

또 하나 아래 seq_along 함수는 벡터 데이터의 원소의 갯수까지 1 부터 하나씩 증가하는 일련번호를 만드는 함수입니다.

아래에서 보시는 바와 같이 flowlist 의 원소의 갯수인 20 까지 1 부터 하나씩 증가하는 일련번호를 만들어 주고 있습니다.

```
seq_along(flowlist)
```

```
##  [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20
```

이러한 사전지식을 가지고 아래와 같이 assign 함수를 이용하여 데이터 파일을 읽어 들여, 분석용 데이터 객체를 만들어 보도록 하겠습니다.

아래의 명령어 표현은 flowlist 의 1 번째 원소와 같은 파일명을 가진 원소를 읽어들여 flow_nm 의 첫번째 원소이름을 가진 객체로 저장하고, 또 2 번, 3 번 해서 20 번째까지 반복적으로 시행하라는 의미입니다.

R 에 익숙치 않으신 분들에게는 조금 어려울 수 있습니다만, 이런 분들은 저희 빅데이터 캠퍼스 분석실 내에 비치된 R 관련 도서들을 보시고, 활용 방법을 충분히 공부하여 주시기 바랍니다.

마지막으로 아래 명령어 표현에서 read.csv 함수의 인자중에 sep 는 파일의 컬럼 항목을 구분하는 구분자를 무엇으로 할지를 지정하는 인자(Argument)를 말합니다. 흔히 CSV 는 구분자(Separator)로 콤마(Comma, ",")를 사용합니다만, 필요에 따라 구분자로 탭문자를 사용하기도 합니다. 이러한 탭문자를 구분자로 사용된 파일을 읽어들인다는 점을 강조하기 위해서는 구분자 표시를 "\t"로 표시해 주시면 됩니다.

R 은 데이터 파일을 읽어들이다가 탭문자("\t")를 만나면 컬럼을 나누어서 데이터 객체로 만들어 주는 것입니다.

```
for (iVar in seq_along(flowlist)) {
  assign(flow_nm[iVar], read.csv(paste0("./flowdata/",flowlist[iVar]),
                                 fileEncoding = "CP949",
                                 encoding = "CP949",
                                 sep = "\t"))
}
```

작업후에 여러분들이 원하는 대로, 데이터 객체 들이 잘 생성되었는지 확인하는 명령어가 ls() 함수 입니다.

아래에 보시는 바와 같이 20 개의 데이터가 여러분들이 지정한 변수명으로 메모리상에 잘 올라가 있는 것을 확인하실 수 있습니다.

```
ls()
```

```
##  [1] "biz_group"                "CULTURE_AGE_INFLOW_2014"
##  [3] "CULTURE_AGE_INFLOW_2015"  "CULTURE_TIME_INFLOW_2014"
##  [5] "CULTURE_TIME_INFLOW_2015" "DRUM_AGE_INFLOW_2014"
##  [7] "DRUM_AGE_INFLOW_2015"     "DRUM_TIME_INFLOW_2014"
##  [9] "DRUM_TIME_INFLOW_2015"    "flow_nm"
## [11] "flowlist"                 "HISEOUL_AGE_INFLOW_2014"
## [13] "HISEOUL_AGE_INFLOW_2015"  "HISEOUL_TIME_INFLOW_2014"
## [15] "HISEOUL_TIME_INFLOW_2015" "iVar"
## [17] "KIMJANG_AGE_INFLOW_2014"  "KIMJANG_AGE_INFLOW_2015"
## [19] "KIMJANG_TIME_INFLOW_2014" "KIMJANG_TIME_INFLOW_2015"
## [21] "LAMP_AGE_INFLOW_2014"     "LAMP_AGE_INFLOW_2015"
## [23] "LAMP_TIME_INFLOW_2014"    "LAMP_TIME_INFLOW_2015"
## [25] "revlist"
```

카드 매출 데이터 Loading

카드 매출 데이터 역시 동일한 절차로 읽어 들일 수 있습니다. 다만, 카드 매출 데이터의 파일명을 보시면 "CULTURE_CARD_2014_215474_NEW.txt", "CULTURE_CARD_2015_298037_NEW.txt" 등과 같이 유동인구 분석용 데이터와 다르게 되어 있습니다. 이중에 축제명을 의미하는 CULTURE, 그리고 신용카드 매출 데이터를 의미하는 CARD, 그리고 년도를 구분하는 2014 와 2015 외에는 중요하지 않은 부분으로 판단되므로, 이들 부분을 모두 제외하고, 데이터 객체를 할당하는 것이 좋을 것으로 생각됩니다.

```
revlist <- dir("./revenuedata/")
revlist
```

```
##  [1] "CULTURE_CARD_2014_215474_NEW.txt" "CULTURE_CARD_2015_298037_NEW.txt"
##  [3] "DRUM_CARD_2014_106822_NEW.txt"    "DRUM_CARD_2015_135082_NEW.txt"
##  [5] "HISEOUL_CARD_2014_436877_NEW.txt" "HISEOUL_CARD_2015_517691_NEW.txt"
##  [7] "KIMJANG_CARD_2014_349363_NEW.txt" "KIMJANG_CARD_2015_431704_NEW.txt"
##  [9] "LAMP_CARD_2014_753748_NEW.txt"    "LAMP_CARD_2015_911871_NEW.txt"
```

아래 코드를 보시면 파일명 리스트의 이름과 문자열 추출의 마지막 글자 위치가 유동인구 데이터 로딩 코드에 비하여 변동되었습니다.

그 아래에 출력된 결과를 보시면, 정확히 여러분들이 원하는 부분까지만 문자열이 추출된 것을 확인하실 수 있습니다.

```
rev_nm <- substr(revlist,1,nchar(revlist)-15)
rev_nm
```

```
##  [1] "CULTURE_CARD_2014" "CULTURE_CARD_2015" "DRUM_CARD_2014"
##  [4] "DRUM_CARD_2015"    "HISEOUL_CARD_2014" "HISEOUL_CARD_2015"
##  [7] "KIMJANG_CARD_2014" "KIMJANG_CARD_2015" "LAMP_CARD_2014"
## [10] "LAMP_CARD_2015"
```

이하의 진행은 유동인구의 경우와 같습니다.

```
for (iVar in seq_along(revlist)) {
  assign(rev_nm[iVar], read.csv(paste0("./revenuedata/",revlist[iVar]),
                                fileEncoding = "CP949",
```

```
                                    encoding = "CP949",
                                    sep = "\t"))
}
ls()
```

```
##  [1] "biz_group"                "CULTURE_AGE_INFLOW_2014"
##  [3] "CULTURE_AGE_INFLOW_2015"  "CULTURE_CARD_2014"
##  [5] "CULTURE_CARD_2015"        "CULTURE_TIME_INFLOW_2014"
##  [7] "CULTURE_TIME_INFLOW_2015" "DRUM_AGE_INFLOW_2014"
##  [9] "DRUM_AGE_INFLOW_2015"     "DRUM_CARD_2014"
## [11] "DRUM_CARD_2015"           "DRUM_TIME_INFLOW_2014"
## [13] "DRUM_TIME_INFLOW_2015"    "flow_nm"
## [15] "flowlist"                 "HISEOUL_AGE_INFLOW_2014"
## [17] "HISEOUL_AGE_INFLOW_2015"  "HISEOUL_CARD_2014"
## [19] "HISEOUL_CARD_2015"        "HISEOUL_TIME_INFLOW_2014"
## [21] "HISEOUL_TIME_INFLOW_2015" "iVar"
## [23] "KIMJANG_AGE_INFLOW_2014"  "KIMJANG_AGE_INFLOW_2015"
## [25] "KIMJANG_CARD_2014"        "KIMJANG_CARD_2015"
## [27] "KIMJANG_TIME_INFLOW_2014" "KIMJANG_TIME_INFLOW_2015"
## [29] "LAMP_AGE_INFLOW_2014"     "LAMP_AGE_INFLOW_2015"
## [31] "LAMP_CARD_2014"           "LAMP_CARD_2015"
## [33] "LAMP_TIME_INFLOW_2014"    "LAMP_TIME_INFLOW_2015"
## [35] "rev_nm"                   "revlist"
```

데이터 정제

R 에서 테이블로 형태로 된 데이터 구조 (이를 데이터 프레임이라고 칭함)를 활용하기 위해서는 각 테이블의 열(Column)에 이름을 붙여서 일종의 변수와 같이 사용하는데, 각 데이터 마다 동일하게 열의 이름이 지정되어 있는지 컬럼(변수라고 도 칭함)은 몇개인지를 확인해 해둘 필요가 있습니다.

왜냐하면, 이런 열의 수가 틀리거나, 동일한 데이터 컬럼을 가지고 있음에도 이름이 다를 경우는 분석시에 오류가 발생할 수도 있기 때문입니다.

위에서 여러개의 데이터를 벡터내에 저장된 문자열이름으로 저장할 때는 assign()함수를 사용하였습니다만, 문자열을 통해 메모리 상에 로드된 데이터 객체를 지정하기 위해서는 get() 함수를 사용하여야 합니다.

```
for (iVar in flow_nm) {
  cat(iVar,"->", dim(get(iVar))[1],"행",dim(get(iVar))[2],"열", "\n")
}
```

```
## CULTURE_AGE_INFLOW_2014 -> 419 행 41 열
## CULTURE_AGE_INFLOW_2015 -> 40652 행 40 열
## CULTURE_TIME_INFLOW_2014 -> 419 행 30 열
## CULTURE_TIME_INFLOW_2015 -> 40741 행 30 열
## DRUM_AGE_INFLOW_2014 -> 417 행 41 열
## DRUM_AGE_INFLOW_2015 -> 27979 행 39 열
## DRUM_TIME_INFLOW_2014 -> 417 행 30 열
## DRUM_TIME_INFLOW_2015 -> 27676 행 28 열
## HISEOUL_AGE_INFLOW_2014 -> 1049 행 41 열
## HISEOUL_AGE_INFLOW_2015 -> 48606 행 41 열
## HISEOUL_TIME_INFLOW_2014 -> 1049 행 30 열
## HISEOUL_TIME_INFLOW_2015 -> 59998 행 30 열
## KIMJANG_AGE_INFLOW_2014 -> 628 행 41 열
## KIMJANG_AGE_INFLOW_2015 -> 630 행 39 열
## KIMJANG_TIME_INFLOW_2014 -> 630 행 30 열
## KIMJANG_TIME_INFLOW_2015 -> 630 행 30 열
## LAMP_AGE_INFLOW_2014 -> 1888 행 41 열
## LAMP_AGE_INFLOW_2015 -> 1890 행 39 열
## LAMP_TIME_INFLOW_2014 -> 1889 행 30 열
## LAMP_TIME_INFLOW_2015 -> 1890 행 30 열
```

위의 실행결과를 보시면 우려했던 바와 같이 열의 갯수가 통일이 되어 있지 않습니다. 프로그래밍을 통해 과업을 해결하기 위해서는 이런 부분을 사전에 잘 정리하는 것이 필요합니다. 이러한 데이터 전처리가 가장 시간과 노력이 많이 들게 됩니다.

이부분을 원하는 형태로 정리되지 않으면 분석은 생각조차 할 수 없기 때문입니다.

그럼 마지막으로 컬럼명 (변수명이라고도 합니다.)은 어떻게 되어 있는지 확인을 해보겠습니다.

데이터의 유형이 연령대별 유동인구와 시간대별 유동인구, 2 가지 유형으로 구분이 되기 때문에 두개 유형을 나눌 수 있도록 객체명을 가지고 구분하도록 하겠습니다. 이럴 때 사용하는 함수가 grep() 함수입니다. 이 함수를 활용하여 객체명에 AGE 와 TIME 이 들어가

있는 객체명의 자리번호를 추출하고, 이들을 따로 분리해낸 다음, 각각 유형들의 객체명을 통일하는 순서로 업무를 진행합니다.

```
flow_age_idx <- grep("AGE", flow_nm)
flow_time_idx <- grep("TIME", flow_nm )
flow_nm_age <- flow_nm[flow_age_idx]
flow_nm_age
```

```
##  [1] "CULTURE_AGE_INFLOW_2014" "CULTURE_AGE_INFLOW_2015"
##  [3] "DRUM_AGE_INFLOW_2014"    "DRUM_AGE_INFLOW_2015"
##  [5] "HISEOUL_AGE_INFLOW_2014" "HISEOUL_AGE_INFLOW_2015"
##  [7] "KIMJANG_AGE_INFLOW_2014" "KIMJANG_AGE_INFLOW_2015"
##  [9] "LAMP_AGE_INFLOW_2014"    "LAMP_AGE_INFLOW_2015"
```

```
flow_nm_time <- flow_nm[flow_time_idx]
flow_nm_time
```

```
##  [1] "CULTURE_TIME_INFLOW_2014" "CULTURE_TIME_INFLOW_2015"
##  [3] "DRUM_TIME_INFLOW_2014"    "DRUM_TIME_INFLOW_2015"
##  [5] "HISEOUL_TIME_INFLOW_2014" "HISEOUL_TIME_INFLOW_2015"
##  [7] "KIMJANG_TIME_INFLOW_2014" "KIMJANG_TIME_INFLOW_2015"
##  [9] "LAMP_TIME_INFLOW_2014"    "LAMP_TIME_INFLOW_2015"
```

먼저 연령대별 유동인구 현황의 변수명을 비교해 보겠습니다.

```
for (iVar in flow_nm_age) {
  cat(iVar,"->", names(get(iVar)), "\n")
}
```

```
## CULTURE_AGE_INFLOW_2014 -> EVENT SECTOR DATE INFLOW_CD INFLOW_NM1 INFLOW_N
M2 INFLOW_NM3 INFLOW_NM4 SUM SUM_M SUM_F M_0004 M_0509 M_1014 M_1519 M_2024 M
_2529 M_3034 M_3539 M_4044 M_4549 M_5054 M_5559 M_6064 M_6569 M_70U F_0004 F_
0509 F_1014 F_1519 F_2024 F_2529 F_3034 F_3539 F_4044 F_4549 F_5054 F_5559 F_
6064 F_6569 F_70U
## CULTURE_AGE_INFLOW_2015 -> EVENT SECTOR DATE INFLOW_CD INFLOW_NM1 INFLOW_N
M2 INFLOW_NM3 SUM SUM_M SUM_F M_0004 M_0509 M_1014 M_1519 M_2024 M_2529 M_303
4 M_3539 M_4044 M_4549 M_5054 M_5559 M_6064 M_6569 M_70U F_0004 F_0509 F_1014
 F_1519 F_2024 F_2529 F_3034 F_3539 F_4044 F_4549 F_5054 F_5559 F_6064 F_6569
 F_70U
## DRUM_AGE_INFLOW_2014 -> EVENT SECTOR DATE INFLOW_CD INFLOW_NM1 INFLOW_NM2
INFLOW_NM3 INFLOW_NM4 SUM SUM_M SUM_F M_0004 M_0509 M_1014 M_1519 M_2024 M_25
29 M_3034 M_3539 M_4044 M_4549 M_5054 M_5559 M_6064 M_6569 M_70U F_0004 F_050
9 F_1014 F_1519 F_2024 F_2529 F_3034 F_3539 F_4044 F_4549 F_5054 F_5559 F_606
4 F_6569 F_70U
## DRUM_AGE_INFLOW_2015 -> DATE INFLOW_CD INFLOW_NM1 INFLOW_NM2 INFLOW_NM3 IN
FLOW_NM4 SUM SUM_M SUM_F M_0004 M_0509 M_1014 M_1519 M_2024 M_2529 M_3034 M_3
539 M_4044 M_4549 M_5054 M_5559 M_6064 M_6569 M_70U F_0004 F_0509 F_1014 F_15
19 F_2024 F_2529 F_3034 F_3539 F_4044 F_4549 F_5054 F_5559 F_6064 F_6569 F_70
U
```

```
## HISEOUL_AGE_INFLOW_2014 -> EVENT SECTOR DATE INFLOW_CD INFLOW_NM1 INFLOW_N
M2 INFLOW_NM3 INFLOW_NM4 SUM SUM_M SUM_F M_0004 M_0509 M_1014 M_1519 M_2024 M
_2529 M_3034 M_3539 M_4044 M_4549 M_5054 M_5559 M_6064 M_6569 M_70U F_0004 F_
0509 F_1014 F_1519 F_2024 F_2529 F_3034 F_3539 F_4044 F_4549 F_5054 F_5559 F_
6064 F_6569 F_70U
## HISEOUL_AGE_INFLOW_2015 -> EVENT SECTOR DATE INFLOW_CD INFLOW_NM1 INFLOW_N
M2 INFLOW_NM3 INFLOW_NM4 SUM SUM_M SUM_F M_0004 M_0509 M_1014 M_1519 M_2024 M
_2529 M_3034 M_3539 M_4044 M_4549 M_5054 M_5559 M_6064 M_6569 M_70U F_0004 F_
0509 F_1014 F_1519 F_2024 F_2529 F_3034 F_3539 F_4044 F_4549 F_5054 F_5559 F_
6064 F_6569 F_70U
## KIMJANG_AGE_INFLOW_2014 -> EVENT SECTOR DATE INFLOW_CD INFLOW_NM1 INFLOW_N
M2 INFLOW_NM3 INFLOW_NM4 SUM SUM_M SUM_F M_0004 M_0509 M_1014 M_1519 M_2024 M
_2529 M_3034 M_3539 M_4044 M_4549 M_5054 M_5559 M_6064 M_6569 M_70U F_0004 F_
0509 F_1014 F_1519 F_2024 F_2529 F_3034 F_3539 F_4044 F_4549 F_5054 F_5559 F_
6064 F_6569 F_70U
## KIMJANG_AGE_INFLOW_2015 -> EVENT SECTOR DATE INFLOW_CD INFLOW_NM1 INFLOW_N
M2 SUM SUM_M SUM_F M_0004 M_0509 M_1014 M_1519 M_2024 M_2529 M_3034 M_3539 M_
4044 M_4549 M_5054 M_5559 M_6064 M_6569 M_70U F_0004 F_0509 F_1014 F_1519 F_2
024 F_2529 F_3034 F_3539 F_4044 F_4549 F_5054 F_5559 F_6064 F_6569 F_70U
## LAMP_AGE_INFLOW_2014 -> EVENT SECTOR DATE INFLOW_CD INFLOW_NM1 INFLOW_NM2
INFLOW_NM3 INFLOW_NM4 SUM SUM_M SUM_F M_0004 M_0509 M_1014 M_1519 M_2024 M_25
29 M_3034 M_3539 M_4044 M_4549 M_5054 M_5559 M_6064 M_6569 M_70U F_0004 F_050
9 F_1014 F_1519 F_2024 F_2529 F_3034 F_3539 F_4044 F_4549 F_5054 F_5559 F_606
4 F_6569 F_70U
## LAMP_AGE_INFLOW_2015 -> EVENT SECTOR DATE INFLOW_CD INFLOW_NM1 INFLOW_NM2
SUM SUM_M SUM_F M_0004 M_0509 M_1014 M_1519 M_2024 M_2529 M_3034 M_3539 M_404
4 M_4549 M_5054 M_5559 M_6064 M_6569 M_70U F_0004 F_0509 F_1014 F_1519 F_2024
 F_2529 F_3034 F_3539 F_4044 F_4549 F_5054 F_5559 F_6064 F_6569 F_70U
```

각각의 변수명은 다음과 같은 의미로 판단됩니다. 컬럼의 변수명을 분석가외 다른 사람이 볼 때에도 이해가 될 수 있도록 잘 정리를 해주신 덕분에 의미를 파악하기 쉬웠습니다.

1) EVENT : 축제명 코드
   - LAMP : 서울 빛초롱 축제
   - CULTURE : 서울 문화의 밤
   - DRUM : 서울 드럼 페스티벌
   - HISEOUL : 하이서울페스티벌
   - KIMJANG : 서울 김장문화제
2) SECTOR : 축제 대상 지역을 섹터로 나누어 구분한 번호
3) DATE : 유동인구를 추출한 날짜
4) INFLOW_CD : 축제 방문객의 거주지역 코드 (이곳에서 축제를 방문했다는 의미)

5) INFLOW_NM : 축제 방문객의 거주지역명(1: 특별시/도 구분, 2:거주지 구명)

6) SUM : 연령대별 유동인구의 총합

7) SUM_M : 연령대별 남성 유동인구의 합

8) SUM_F : 연령대별 여성 유동인구의 합

9) M_0004 : 남성(Male) 0 세 ~ 5 세미만 유동인구 수

10) M_0509 : 남성(Male) 5 세 ~ 10 세미만 유동인구 수

11) M_1014 : 끝의 4 자리 숫자를 연령대로 해석하여 활용. (기타 변수 설명 생략)

12) M_70U : 남성(Male) 70 세 이상 유동인구 수

13) F_0004 : 여성(Female) 0 세 ~ 5 세미만 유동인구 수

14) F_0509 : 여성(Female) 5 세 ~ 10 세미만 유동인구 수

15) F_1014 : 끝의 4 자리 숫자를 연령대로 해석하여 활용. (기타 변수 설명 생략)

16) F_70U : 여성(Female) 70 세 이상 유동인구 수

다음은 시간대별 유동인구 현황의 변수명을 비교해 보겠습니다.

```
for (iVar in flow_nm_time) {
  cat(iVar,"->", names(get(iVar)), "\n\n")
}
```

```
## CULTURE_TIME_INFLOW_2014 -> EVENT SECTOR DATE INFLOW_CD INFLOW_NM SUM TIME
_0 TIME_1 TIME_2 TIME_3 TIME_4 TIME_5 TIME_6 TIME_7 TIME_8 TIME_9 TIME_10 TIM
E_11 TIME_12 TIME_13 TIME_14 TIME_15 TIME_16 TIME_17 TIME_18 TIME_19 TIME_20
TIME_21 TIME_22 TIME_23
##
## CULTURE_TIME_INFLOW_2015 -> EVENT SECTOR DATE INFLOW_CD INFLOW SUM X0 X1 X
2 X3 X4 X5 X6 X7 X8 X9 X10 X11 X12 X13 X14 X15 X16 X17 X18 X19 X20 X21 X22 X2
3
##
## DRUM_TIME_INFLOW_2014 -> EVENT SECTOR DATE INFLOW_CD INFLOW_NM SUM TIME_0
TIME_1 TIME_2 TIME_3 TIME_4 TIME_5 TIME_6 TIME_7 TIME_8 TIME_9 TIME_10 TIME_1
1 TIME_12 TIME_13 TIME_14 TIME_15 TIME_16 TIME_17 TIME_18 TIME_19 TIME_20 TIM
E_21 TIME_22 TIME_23
##
## DRUM_TIME_INFLOW_2015 -> DATE INFLOW_CD INFLOW SUM X0 X1 X2 X3 X4 X5 X6 X7
 X8 X9 X10 X11 X12 X13 X14 X15 X16 X17 X18 X19 X20 X21 X22 X23
##
## HISEOUL_TIME_INFLOW_2014 -> EVENT SECTOR DATE INFLOW_CD INFLOW_NM SUM TIME
_0 TIME_1 TIME_2 TIME_3 TIME_4 TIME_5 TIME_6 TIME_7 TIME_8 TIME_9 TIME_10 TIM
```

```
E_11 TIME_12 TIME_13 TIME_14 TIME_15 TIME_16 TIME_17 TIME_18 TIME_19 TIME_20
TIME_21 TIME_22 TIME_23
##
## HISEOUL_TIME_INFLOW_2015 -> EVENT SECTOR DATE INFLOW_CD INFLOW SUM X0 X1 X
2 X3 X4 X5 X6 X7 X8 X9 X10 X11 X12 X13 X14 X15 X16 X17 X18 X19 X20 X21 X22 X2
3
##
## KIMJANG_TIME_INFLOW_2014 -> EVENT SECTOR DATE INFLOW_CD INFLOW_NM SUM TIME
_0 TIME_1 TIME_2 TIME_3 TIME_4 TIME_5 TIME_6 TIME_7 TIME_8 TIME_9 TIME_10 TIM
E_11 TIME_12 TIME_13 TIME_14 TIME_15 TIME_16 TIME_17 TIME_18 TIME_19 TIME_20
TIME_21 TIME_22 TIME_23
##
## KIMJANG_TIME_INFLOW_2015 -> EVENT SECTOR DATE INFLOW_CD INFLOW_NM SUM TIME
_0 TIME_1 TIME_2 TIME_3 TIME_4 TIME_5 TIME_6 TIME_7 TIME_8 TIME_9 TIME_10 TIM
E_11 TIME_12 TIME_13 TIME_14 TIME_15 TIME_16 TIME_17 TIME_18 TIME_19 TIME_20
TIME_21 TIME_22 TIME_23
##
## LAMP_TIME_INFLOW_2014 -> EVENT SECTOR DATE INFLOW_CD INFLOW_NM SUM TIME_0
TIME_1 TIME_2 TIME_3 TIME_4 TIME_5 TIME_6 TIME_7 TIME_8 TIME_9 TIME_10 TIME_1
1 TIME_12 TIME_13 TIME_14 TIME_15 TIME_16 TIME_17 TIME_18 TIME_19 TIME_20 TIM
E_21 TIME_22 TIME_23
##
## LAMP_TIME_INFLOW_2015 -> EVENT SECTOR DATE INFLOW_CD INFLOW_NM SUM TIME_0
TIME_1 TIME_2 TIME_3 TIME_4 TIME_5 TIME_6 TIME_7 TIME_8 TIME_9 TIME_10 TIME_1
1 TIME_12 TIME_13 TIME_14 TIME_15 TIME_16 TIME_17 TIME_18 TIME_19 TIME_20 TIM
E_21 TIME_22 TIME_23
```

위의 결과를 보면, DRUM(서울 드럼페스티벌)의 2015 년 데이터에 EVENT 명과 SECTOR 번호가 생략되어 있습니다. SECTOR 번호등 중요한 정보가 없는 것으로 보여, 일단 DRUM 은 시간대별 유동인구 분석에서 제외하는 것이 좋을 것 같습니다.

```
flow_nm_time2 <- flow_nm_time[-c(3,4)]
```

서울 드럼페스티벌을 제외하고, EVENT, SECTOR, DATE, INFLOW_CD, INFLOW_NM, SUM, 그리고 기간대별 유동인구의 순서인 것으로 파악됩니다만, 각각 데이터객체의 컬럼명이 제각각입니다. 이런 부분들을 일치시켜 주어야 합니다.

연령대별 유동인구의 컬럼명을 기준으로 시간대는 "T_1" ~ "T_23" 등으로 맞추도록 하겠습니다.

```
for (iVar in flow_nm_time2) {
  dummy <- get(iVar)
  names(dummy) <- c("EVENT", "SECTOR", "DATE", "INFLOW_CD", "INFLOW_NM", "SUM
",
                          paste0("T_",0:23))
```

```
  assign(iVar, dummy)
  cat(iVar,"->", names(get(iVar)), "\n\n")
}
```

```
## CULTURE_TIME_INFLOW_2014 -> EVENT SECTOR DATE INFLOW_CD INFLOW_NM SUM T_0
T_1 T_2 T_3 T_4 T_5 T_6 T_7 T_8 T_9 T_10 T_11 T_12 T_13 T_14 T_15 T_16 T_17 T
_18 T_19 T_20 T_21 T_22 T_23
##
## CULTURE_TIME_INFLOW_2015 -> EVENT SECTOR DATE INFLOW_CD INFLOW_NM SUM T_0
T_1 T_2 T_3 T_4 T_5 T_6 T_7 T_8 T_9 T_10 T_11 T_12 T_13 T_14 T_15 T_16 T_17 T
_18 T_19 T_20 T_21 T_22 T_23
##
## HISEOUL_TIME_INFLOW_2014 -> EVENT SECTOR DATE INFLOW_CD INFLOW_NM SUM T_0
T_1 T_2 T_3 T_4 T_5 T_6 T_7 T_8 T_9 T_10 T_11 T_12 T_13 T_14 T_15 T_16 T_17 T
_18 T_19 T_20 T_21 T_22 T_23
##
## HISEOUL_TIME_INFLOW_2015 -> EVENT SECTOR DATE INFLOW_CD INFLOW_NM SUM T_0
T_1 T_2 T_3 T_4 T_5 T_6 T_7 T_8 T_9 T_10 T_11 T_12 T_13 T_14 T_15 T_16 T_17 T
_18 T_19 T_20 T_21 T_22 T_23
##
## KIMJANG_TIME_INFLOW_2014 -> EVENT SECTOR DATE INFLOW_CD INFLOW_NM SUM T_0
T_1 T_2 T_3 T_4 T_5 T_6 T_7 T_8 T_9 T_10 T_11 T_12 T_13 T_14 T_15 T_16 T_17 T
_18 T_19 T_20 T_21 T_22 T_23
##
## KIMJANG_TIME_INFLOW_2015 -> EVENT SECTOR DATE INFLOW_CD INFLOW_NM SUM T_0
T_1 T_2 T_3 T_4 T_5 T_6 T_7 T_8 T_9 T_10 T_11 T_12 T_13 T_14 T_15 T_16 T_17 T
_18 T_19 T_20 T_21 T_22 T_23
##
## LAMP_TIME_INFLOW_2014 -> EVENT SECTOR DATE INFLOW_CD INFLOW_NM SUM T_0 T_1
 T_2 T_3 T_4 T_5 T_6 T_7 T_8 T_9 T_10 T_11 T_12 T_13 T_14 T_15 T_16 T_17 T_18
 T_19 T_20 T_21 T_22 T_23
##
## LAMP_TIME_INFLOW_2015 -> EVENT SECTOR DATE INFLOW_CD INFLOW_NM SUM T_0 T_1
 T_2 T_3 T_4 T_5 T_6 T_7 T_8 T_9 T_10 T_11 T_12 T_13 T_14 T_15 T_16 T_17 T_18
 T_19 T_20 T_21 T_22 T_23
```

마지막으로 매출관련 데이터를 리뷰해보도록 하겠습니다. 방법은 위의 유동인구 데이터 검증과 유사한 절차로 진행합니다. 아래의 코드로 축제별, 연도별 데이터가 동일한 컬럼명으로 제대로 작성되어 있습니다만, 각각의 데이터에 EVENT(축제명) 컬럼이 생략되어 있습니다. 이 부분을 추가할 필요가 있을 것 같습니다.

```
for (iVar in rev_nm) {
  cat(iVar,"->", names(get(iVar)), "\n\n")
}
```

```
## CULTURE_CARD_2014 -> SECTOR DATE CCD SEX AGE INFLOW TIME MONEY COUNT
##
## CULTURE_CARD_2015 -> SECTOR DATE CCD SEX AGE INFLOW TIME MONEY COUNT
##
## DRUM_CARD_2014 -> SECTOR DATE CCD SEX AGE INFLOW TIME MONEY COUNT
##
## DRUM_CARD_2015 -> SECTOR DATE CCD SEX AGE INFLOW TIME MONEY COUNT
##
## HISEOUL_CARD_2014 -> SECTOR DATE CCD SEX AGE INFLOW TIME MONEY COUNT
##
## HISEOUL_CARD_2015 -> SECTOR DATE CCD SEX AGE INFLOW TIME MONEY COUNT
##
## KIMJANG_CARD_2014 -> SECTOR DATE CCD SEX AGE INFLOW TIME MONEY COUNT
##
## KIMJANG_CARD_2015 -> SECTOR DATE CCD SEX AGE INFLOW TIME MONEY COUNT
##
## LAMP_CARD_2014 -> SECTOR DATE CCD SEX AGE INFLOW TIME MONEY COUNT
##
## LAMP_CARD_2015 -> SECTOR DATE CCD SEX AGE INFLOW TIME MONEY COUNT
```

```
library(stringr)
event_nm <- str_extract(rev_nm,"^([A-Z]*)")
```

```
for (iVar in seq_along(rev_nm)) {
  dummy <- get(rev_nm[iVar])
  dummy$EVENT <- event_nm[iVar]
  dummy <- dummy[,c(10,1:9)]
  assign(rev_nm[iVar], dummy)
  cat(rev_nm[iVar],"->", names(get(rev_nm[iVar])), "\n\n")
}
```

```
## CULTURE_CARD_2014 -> EVENT SECTOR DATE CCD SEX AGE INFLOW TIME MONEY COUNT

##
## CULTURE_CARD_2015 -> EVENT SECTOR DATE CCD SEX AGE INFLOW TIME MONEY COUNT

##
## DRUM_CARD_2014 -> EVENT SECTOR DATE CCD SEX AGE INFLOW TIME MONEY COUNT
##
## DRUM_CARD_2015 -> EVENT SECTOR DATE CCD SEX AGE INFLOW TIME MONEY COUNT
##
## HISEOUL_CARD_2014 -> EVENT SECTOR DATE CCD SEX AGE INFLOW TIME MONEY COUNT

##
## HISEOUL_CARD_2015 -> EVENT SECTOR DATE CCD SEX AGE INFLOW TIME MONEY COUNT

##
## KIMJANG_CARD_2014 -> EVENT SECTOR DATE CCD SEX AGE INFLOW TIME MONEY COUNT
```

```
##
## KIMJANG_CARD_2015 -> EVENT SECTOR DATE CCD SEX AGE INFLOW TIME MONEY COUNT

##
## LAMP_CARD_2014 -> EVENT SECTOR DATE CCD SEX AGE INFLOW TIME MONEY COUNT
##
## LAMP_CARD_2015 -> EVENT SECTOR DATE CCD SEX AGE INFLOW TIME MONEY COUNT
```

각각의 컬럼명은 다음과 같습니다. 주의해야할 사항은 유입지가 코드로만 되어 있어서, 어느지역인지 알기 위해서는 행정동 코드 등을 중심으로 구분할 필요가 있습니다.

1) EVENT : 축제명 코드
   - LAMP : 서울 빛초롱 축제
   - CULTURE : 서울 문화의 밤
   - DRUM : 서울 드럼 페스티벌
   - HISEOUL : 하이서울페스티벌
   - KIMJANG : 서울 김장문화제
2) SECTOR : 축제 대상 지역을 섹터로 나누어 구분한 번호
3) DATE : 카드매출을 추출한 날짜
4) CCD : 매출 발생업종코드
5) SEX: 남여 구분
6) AGE : 연령대 구분
7) TIME : 시간대 구분
8) MONEY : 매출금액 (단위: 원)
9) COUNT : 매출발생 건수

탐색적 데이터 분석(Explorative Data Analysis)

5 개 축제중 서울드럼 페스티벌은 연령대별 유동인구 2015 년 데이터에 오류가 있는 것으로 판단되는 바, 제외하고 나머지 4 개 축제 중 하이서울 페스티벌을 위주로 분석을

해보도록 하겠습니다. 데이터의 구조를 모두 통일시켜 놓았기 때문에 동일한 분석기법을 다른 축제에 적용하는데 큰 무리가 없을 것으로 판단됩니다.

먼저 하이서울 페스티벌의 데이터가 어떤식으로 구성되어 있는지 보도록 하겠습니다. 데이터 집계를 위한 R 의 패키지인 dplyr 을 사용하여 탐색적 데이터 분석을 진행해보도록 하겠습니다.

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
hi_c14 <- HISEOUL_CARD_2014
synop14 <- hi_c14 %>%
  group_by(DATE) %>%
  summarize(sumcount=sum(COUNT), sales=sum(MONEY))
visit_14 <- as.data.frame(synop14)
```

요일구분을 위해 strptime()과 strftime()함수를 사용하여 요일을 추출하고, 축제기간 여부(1 이 축제기간을 의미함)를 별도로 지정해서 추후의 분석에 대비해 둡니다.

```
visit_14$wday <- format(strptime(visit_14$DATE, "%Y%m%d"), "%A")
visit_14$year <- format(strptime(visit_14$DATE, "%Y%m%d"), "%Y")
visit_14$mday <- format(strptime(visit_14$DATE, "%Y%m%d"), "%m%d")
visit_14$festival <- ifelse(visit_14$DATE >= 20141001 &
                            visit_14$DATE <= 20141004, 1,0)
visit_14
```

```
##        DATE sumcount     sales   wday year mday festival
## 1  20140916    16296 292018531 화요일 2014 0916        0
## 2  20140917    16737 302316063 수요일 2014 0917        0
## 3  20140918    16995 319402183 목요일 2014 0918        0
## 4  20140919    17722 338660105 금요일 2014 0919        0
## 5  20140920    11766 318246200 토요일 2014 0920        0
## 6  20140921     9048 155519414 일요일 2014 0921        0
```

```
## 7  20140922    15897 279450360 월요일 2014 0922        0
## 8  20140923    16404 292417025 화요일 2014 0923        0
## 9  20140924    15760 292132404 수요일 2014 0924        0
## 10 20140925    16723 316571924 목요일 2014 0925        0
## 11 20140926    16991 325733403 금요일 2014 0926        0
## 12 20140927    11389 266210560 토요일 2014 0927        0
## 13 20140928     7884 145984598 일요일 2014 0928        0
## 14 20140929    14840 276019221 월요일 2014 0929        0
## 15 20140930    16981 342670967 화요일 2014 0930        0
## 16 20141001    16975 310293679 수요일 2014 1001        1
## 17 20141002    16332 324037876 목요일 2014 1002        1
## 18 20141003    10100 212393418 금요일 2014 1003        1
## 19 20141004    10265 252927751 토요일 2014 1004        1
## 20 20141005     8512 145294357 일요일 2014 1005        0
## 21 20141006    15762 273420949 월요일 2014 1006        0
## 22 20141007    16279 296123835 화요일 2014 1007        0
## 23 20141008    16876 312927023 수요일 2014 1008        0
## 24 20141009    10943 206767970 목요일 2014 1009        0
## 25 20141010    16485 283580150 금요일 2014 1010        0
## 26 20141011    10552 232888263 토요일 2014 1011        0
## 27 20141012     8344 158692921 일요일 2014 1012        0
## 28 20141013    15764 300592695 월요일 2014 1013        0
## 29 20141014    15917 277894285 화요일 2014 1014        0
## 30 20141015    16333 288004384 수요일 2014 1015        0
## 31 20141016    15958 295625984 목요일 2014 1016        0
## 32 20141017    16574 296173864 금요일 2014 1017        0
## 33 20141018    10439 279220469 토요일 2014 1018        0
## 34 20141019     8115 177573100 일요일 2014 1019        0
## 35 20141020    14191 248818094 월요일 2014 1020        0
```

```r
library(dplyr)
hi_c15 <- HISEOUL_CARD_2015
synop15 <- hi_c15 %>%
  group_by(DATE) %>%
  summarize(sumcount=sum(COUNT), sales=sum(MONEY))
```

```r
visit_15 <- as.data.frame(synop15)
visit_15$wday <- format(strptime(visit_15$DATE, "%Y%m%d"), "%A")
visit_15$year <- format(strptime(visit_15$DATE, "%Y%m%d"), "%Y")
visit_15$mday <- format(strptime(visit_15$DATE, "%Y%m%d"), "%m%d")
visit_15$festival <- ifelse(visit_15$DATE >= 20151001 &
                            visit_15$DATE <= 20151004, 1,0)
visit_15
```

```
##        DATE sumcount     sales   wday year mday festival
## 1  20150916    21997 351492676 수요일 2015 0916        0
## 2  20150917    21918 362291410 목요일 2015 0917        0
## 3  20150918    22890 404615752 금요일 2015 0918        0
## 4  20150919    14106 289619523 토요일 2015 0919        0
## 5  20150920    10474 165073216 일요일 2015 0920        0
## 6  20150921    21117 351787353 월요일 2015 0921        0
## 7  20150922    22184 370378223 화요일 2015 0922        0
## 8  20150923    22413 440138157 수요일 2015 0923        0
## 9  20150924    22786 415075547 목요일 2015 0924        0
## 10 20150925    20750 360485809 금요일 2015 0925        0
## 11 20150926     6549 105610403 토요일 2015 0926        0
## 12 20150927     4713  71698631 일요일 2015 0927        0
## 13 20150928     8328 138154099 월요일 2015 0928        0
## 14 20150929    10353 167691071 화요일 2015 0929        0
## 15 20150930    20836 365527379 수요일 2015 0930        0
## 16 20151001    19658 331822915 목요일 2015 1001        1
## 17 20151002    22313 389473696 금요일 2015 1002        1
## 18 20151003    13890 285574133 토요일 2015 1003        1
## 19 20151004    10768 187773204 일요일 2015 1004        1
## 20 20151005    21469 345099977 월요일 2015 1005        0
## 21 20151006    22181 385992742 화요일 2015 1006        0
## 22 20151007    22405 359224903 수요일 2015 1007        0
## 23 20151008    22876 400856102 목요일 2015 1008        0
## 24 20151009    14170 283222122 금요일 2015 1009        0
## 25 20151010    11355 305428915 토요일 2015 1010        0
## 26 20151011     9367 181692867 일요일 2015 1011        0
## 27 20151012    20404 318663574 월요일 2015 1012        0
```

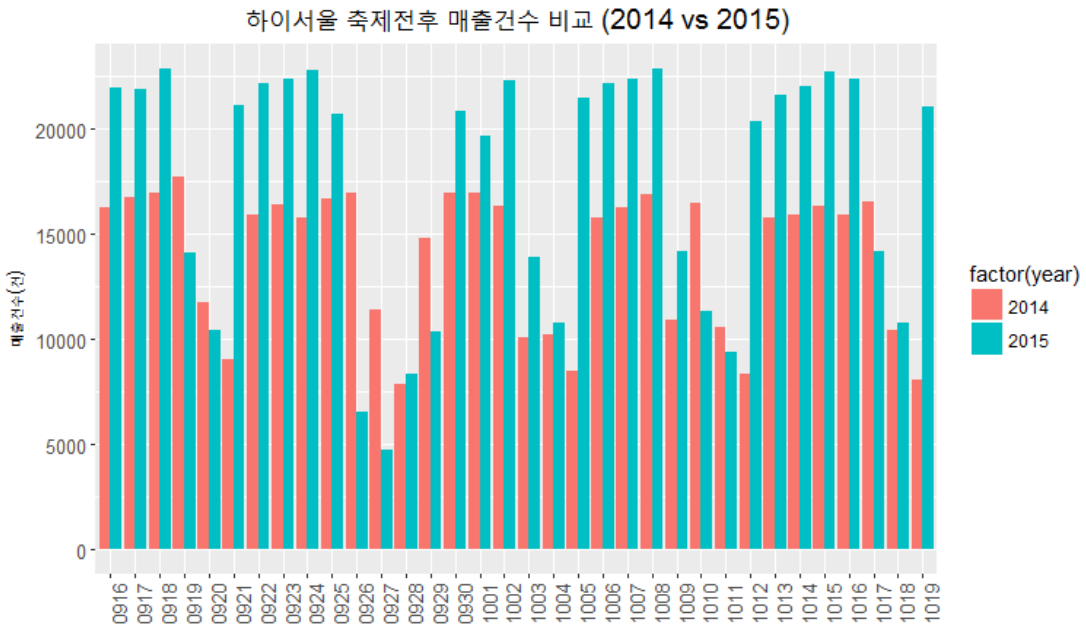
```
## 28 20151013    21644 348277107 화요일 2015 1013        0
## 29 20151014    22070 362786941 수요일 2015 1014        0
## 30 20151015    22719 378876222 목요일 2015 1015        0
## 31 20151016    22385 406850208 금요일 2015 1016        0
## 32 20151017    14200 326244803 토요일 2015 1017        0
## 33 20151018    10761 175230913 일요일 2015 1018        0
## 34 20151019    21073 330923603 월요일 2015 1019        0
```

데이터 시각화 (Visualization)

```
library(ggplot2)
pltdata <- rbind(visit_14[-35,c("year","mday","sumcount")],
                 visit_15[,c("year","mday","sumcount")])

llplt <- ggplot(pltdata, aes(x=factor(mday), y=sumcount, fill=factor(year)))

cplot <- llplt +
  geom_bar(stat="identity", position="dodge") +
  labs(title="하이서울 축제전후 매출건수 비교 (2014 vs 2015)",x="",y="매출건수
(건)") +
  theme(plot.title=element_text(family="Korea1", size=14, hjust=0.5),
        axis.text.x=element_text(family="Korea1",size=10, angle=90, vjust=1,
hjust=0.4),
        axis.title.y=element_text(family="Korea1",size=10),
        axis.title.x=element_text(family="Korea1",size=10),
        axis.text.y=element_text(family="Korea1",size=10),
        strip.text=element_text(family="Korea1",size=10),
        plot.margin = unit(c(0.9,0.1,0.1,0.5), "cm"))
cplot
```



최종 Wrap-Up

본 분석을 위해 사용한 컴퓨터 환경 및 R 의 패키지 내역은 아래와 같습니다.

```
sessionInfo()
```

```
## R version 3.3.2 (2016-10-31)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
## Running under: Windows 8.1 x64 (build 9600)
##
## locale:
## [1] LC_COLLATE=Korean_Korea.949  LC_CTYPE=Korean_Korea.949
## [3] LC_MONETARY=Korean_Korea.949 LC_NUMERIC=C
## [5] LC_TIME=Korean_Korea.949
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
## [1] ggplot2_2.2.0 dplyr_0.5.0   stringr_1.1.0
##
## loaded via a namespace (and not attached):
##  [1] Rcpp_0.12.8      digest_0.6.10    rprojroot_1.1    assertthat_0.1
##  [5] plyr_1.8.4       grid_3.3.2       R6_2.2.0         gtable_0.2.0
```

```
##  [9] DBI_0.5-1        backports_1.0.4  magrittr_1.5     scales_0.4.1
## [13] evaluate_0.10    stringi_1.1.2    lazyeval_0.2.0   rmarkdown_1.3
## [17] labeling_0.3     tools_3.3.2      munsell_0.4.3    yaml_2.1.14
## [21] colorspace_1.3-2 htmltools_0.3.5  knitr_1.15.1     tibble_1.2
```

서울시 빅데이터 캠퍼스를 통해 다양한 사회문제 데이터 분석으로 사회혁신을 유도하는 새로운 인사이트를 발견하고 공유하시기를 바랍니다.

감사합니다.