2018년 6월 28일 4일차

MySQL Workbench 실습

1) E-R 다이어그램 모델 제작 : 논리 설계

> 새로운 Connection 생성 : 스키마는 계정 단위로 공유. Connection 은 동일한 스키마를 가지게 된다.
> Models 탭에 들어가 + 버튼 클릭
> Add Diagram 을 이용해 E-R 다이어그램 생성창에 진입
> 테이블 형식을 만들기 위해 오른편 Open the Table Template Editor 클릭.
> 테이블 이름, 컬럼 이름, 데이터 타입, 기본값, 제약조건을 설정하여 테이블 형식 생성
> E-R 다이어그램 생성창에서 템플릿을 가져와 테이블을 생성
> 테이블간의 PK - FK 관계는 1 : N / N : 1 / N : N 선을 이용하여 컬럼끼리 연결

2) 실제 테이블 생성 : 물리 설계

> E-R 다이어그램 창에서 Database - Forward Engineer 를 이용해 실제 테이블 생성
> 이 때, 코드를 살펴보고 문제가 있으면 수정하여 생성해야 한다.
> 저장한 Connection 으로 가서 Schemas 를 Refresh 하면 생성된 것을 확인할 수 있다.

R 과 R Studio 기본

script - console - evironment 창으로 실습

1) Vector

vec <- c (1, 2, 3, 4)
> 1, 2, 3, 4를 원소로 가지는 벡터를 vec 이름으로 생성

vec[a]
> a 번째 원소에 접근. R 의 Vector 는 시작값이 1이다.

vec[c(T, F, T, F)]
> True 가 입력된 1번째와 3번째 값에만 접근하여 출력
> 즉, 1, 3 만 출력된다

2) Matrix

matrix (c (1 : 12) , nrow = 3)
> 앞의 벡터로 매트릭스를 생성하며, 매트릭스의 행은 3이 된다.
> 1 : 12 는 1부터 12까지 연속적인 등차수열을 의미한다.
> 이 때, 기본적으로 열 단위로 숫자가 출력되게 된다 (세로로 1, 2, 3 ... 순서)

matrix (c (1 : 12) , nrow = 3 , byrow = T)
> 행 단위로 숫자가 출력되게 한다

3) Array

array (1 : 12, dim = c (2, 2, 3))
> 1 부터 12 까지 연속적인 숫자로 3차원 배열을 만든다.
> 2 x 2 짜리 매트릭스를 3개 겹쳐놓는 형태이다

4) List

list (name = c (“ A “ , “ B “) , age = c (20 , 21)

> name 키에 A, B 의 벡터를 , age 키에 20, 21 의 벡터를 가지는 리스트를 생성한다

list01$name[1]
> 위의 list를 list01 이라는 이름으로 정의했을 때, name 키값의 밸류에서
> 첫 번재 인자를 가져오는 것이다

5) Dataframe

df = data.frame(name = c(“A”, “B”, “C”), age = c(1, 2, 3))
>

df[,]
>

6) Package

install.packages(“KoNLP”) or Packages 의 Install 버튼을 이용해 설치
> 한국어 자연어처리 패키지

install.packages(c(“httr”, “rvest”))
> httr : http 와 r 의 합성어로 웹페이지를 가져오는 라이브러리임.
> rvest : harvest 와 r 의 합성어로 웹페이지에서 특정 노드를 추출하는 라이브러리임

httr 은 URL 을 Server 쪽에 요청하는 역할을 한다.
rvest 는

library (library_name) 를 이용하여 라이브러리를 불러와야 한다
이 때는 vector 를 이용해 한 번에 불러오는 것이 불가능하다.

R 을 이용한 웹 크롤링

1) Page URL 가져오기

> 일반적인 URL 이 아닌, Page 와 날짜 정보등이 모두 드러나는 URL 로 가져와야 한다.
> Script 에 url 변수를 만들어 string 형태로 url 정보를 저장한다.

GET (base_url)
> base_url 정보를 이용하여 웹페이지 전체를 가져온다

2) link 정보가 모두 담긴 하나의 div 를 찾아내기

html_news = GET(base_url)
html_news01 = read_html(html_news)
> 먼저 앞에서 GET 으로 가져온 html 파일을 하나의 변수에 넣고, 메타데이터를 제거한다.

html_nodes(html_new01, “div.list_body a”)
> list_body 내의 모든 링크 노드를 가져온다.

link_news01 <- html_attr(link_news, 'href')
> 각각의 링크 노드에서 href 어트리뷰트만 뽑아낸다.

link_news02 <- unique(link_news01)
> 현재는 중복된 링크가 있으므로, 중복된 링크 값들을 제거해 준다.

link_news03 <- grep('news.naver.com', link_news02, value = T)
> 다른 링크들이 존재하므로 이를 걸러내기 위해 grep 을 사용한다.

3) 하나의 본문 내용을 뽑아오기

http_contents <- GET(link_news03[1])
html_contents <- read_html(http_contents)
> 여러개의 링크중 하나만 가져와, http_contents 에 넣고, 이를 html 로 변환하여 가져온다.
> 즉, 본문 내용의 전체 html 을 가져와 html_contents 에 넣는 것이다.

contents_area <- html_nodes(html_contents, 'div#articleBodyContents')
>

4) for 반복문으로 모든 본문 내용 뽑아오기

for (i in 1 : length (link_news03)) { }
> link_news03 의 길이만큼 반복하며, link_news03[i] 에 접근한다.

tip) 여러 줄을 선택한 다음, cmd + i 를 입력하면 자동으로 줄맞춤이 실행된다

text_all = c( )
> for 문 바깥에서 하나의 빈 벡터를 정의한다

text <- html_text(contents_area)
text_all <- c (text_all , text)
> text 변수를 text_all 벡터에 for 문의 반복에 따라 쌓아가는 것이다

5) for 반복문으로 페이지를 진행하며 본문 내용 뽑아오기

base_url 의 마지막 숫자인 페이지 숫자를 삭제하고 저장한다.

```
for ( j in 1 : 10 ) {
        url_news <- paste ( base_url , j , sep = '' )
        …
}
```
> for 문을 1 부터 원하는 페이지 수만큼 진행하면서 앞의 base_url 의 페이지가 들어갈 부분에
> 인덱스인 j 를 paste 함수로 붙여서 크롤링을 진행한다.
> 이 때, paste 함수는 기본적으로 두 값의 구분, 즉 separate 를 띄어쓰기로 설정하므로
> 이 설정을 바꿔주기 위해 sep = '' 을 넣어주도록 한다.

sys.sleep(time = 3)
> 몇몇 페이지는 특정 속도 이상으로 다량의 페이지에 접근할 경우 Block 시키므로
> 이를 피하기 위해 sys.sleep (time = a) 함수로 a 초 만큼의 딜레이를 넣어준다.

6) R 의 저장

코드의 경우 . R 의 형식으로 저장된다.
데이터의 경우 . RDATA 형식으로 저장된다.

즉, 데이터를 뽑아낸 다음, 해당 데이터를 따로 저장해줘야 한다.
데이터 뿐만 아니라