입력과 출력

입력

키보드

- 입력 데이터가 아주 적을 때
  - c() 함수를 이용해 직접 입력
- R에서 제공하는 데이터 편집기 이용
  - edit() 함수 이용 : edit(데이터프레임)으로 켜기
    - 스프레드시트를 사용하듯이 사용가능, 변수 이름/유형 변경이 쉬움
    - 입력내용이 변수로 저장됨 : dataframe <- edit(dataframe) 또는 fix(dataframe)
- 클립보드에서 읽어오기
  - 엑셀의 셀들을 선택해 복사
  - readClipboard()함수 호출
  - 두개 이상의 행/열을 읽어올때는 \t가 포함된 형태로 데이터들을 불러옴
    - read.table() 함수 이용
    - read.table(file="clipboard", sep="\t", header = TRUE)
    - 세퍼레이터를 \t로, 헤더가 있음을 TRUE로 표현
    - text인수도 사용가능(텍스트벡터 형식의 데이터 불러오기) : text = mytxt

파일

- 대량의 데이터를 다룰 때 적합한 방법
- CSV(Comma-Separated Value) 형식 : 데이터 입력 파일 중 가장 보편적으로 사용되는 형식
  - 쉼표를 이용해 데이터요소를 구분
  - 각 라인은 하나의 행, 쉼표로 구분된 데이터 항목들로 구성
  - read.csv() 함수 : 첫번 째 인수(file)로 주어진 CSV 파일을 읽어들여 데이터프레임 생성
    - 지정되지 않으면 해당 CSV에 **헤더가 포함**되어있다고 가정(header = TRUE)
    - 지정되지 않으면 구분자는 콤마(sep =",")
    - 지정되지 않으면 문자를 모두 Factor로 변환
      - stringsAsFactors = FALSE / as.is =TRUE 옵션 이용
- 테이블 형식
  - 각 라인은 하나의 레코드로 구성, 필드들이 구분자에 의해 구별
  - 각 레코드는 동일한 개수의 필드를 가짐
  - read.csv 함수와 유사하지만, read.table이 좀 더 다양한 옵션 제공
    - 지정되지 않으면 **헤더가 없다**고 가정(header = FALSE)
    - 지정되지 않으면 문자를 모두 Factor로 변환

            - stringsAsFactors = FALSE 옵션 이용
        - 지정되지 않으면 구분자는 공백(sep = " ")
        - NA를 결측값으로 간주 -> 다른 문자로 결측값을 나타낼거라면 na.strings에 해당 문자 기입
        - 파일을 읽어들일 때 #으로 시작하는 라인은 **주석**으로 생각
        - row.names 옵션으로 **행이름** 지정가능
        - colClasses 옵션으로 각 열의 클래스 지정가능
    - read.fwf 함수 : 고정된 길이로 데이터 필드를 구분
        - widths 인수 : 각 데이터 필드의 길이 차례대로 지정
            - 음수일시 해당숫자만큼의 열을 건너뜀
            - 0일시 결측값으로 채워짐
        - col.names 인수 : 각 열의 이름 지정
- 데이터 구조가 복잡하거나 불규칙적일 때
    - readLines() 함수 : 라인 단위로 데이터를 읽어와 문자 벡터 반환
        - con 인수 : 읽어들일 파일 이름 지정
        - n 인수 : 읽어들일 라인 개수 지정
    - scan() : readLines 보다 풍부한 기능 제공, 데이터 항목 단위로 데이터 처리
        - file 인수 : 읽어들일 파일 이름 지정
        - what 인수 : 읽어들일 데이터 항목의 유형 지정 -> 반복적 사용
            - ex) numeric(0), integer(0), complex(0), character(0), logical(0)
            - 리스트 형식으로 지정가능 -> 각자의 순서에 맞는 원소에 저장이 됨
        - nlines 인수 : 읽어들일 라인 개수 지정
        - skip 인수 : 읽지않고 건너 뛸 라인의 개수 지정
- 엑셀 파일로부터 직접 읽어들이기 : 패키지 설치 필요
    - openxlsx 패키지의 read.xlsx()
        - xlsxFile 인수 : 엑셀 파일 지정
        - sheet 인수 : 읽어들일 워크시트 번호 지정
        - colNames 인수 : 변수 이름 어떡할지 지정, 기본값은 TRUE(첫 행을 변수 이름으로 사용)

웹사이트

- 웹상의 데이터를 읽어들이는 다양한 방법 제공
- HTTP 뿐만 아니라 FTP 서버도 포함 가능
- read.csv(), read.table()의 파일 인수에 url을 집어넣으면 됨
- download.file() 함수 : 파일을 컴퓨터로 다운로드해 사용
    - url 인수 : URL 지정
    - destfile 인수 : 저장할 파일 이름 지정
    - 바이너리 모드 (mode = "wb")를 지정해 노래를 다운로드 해와서 틀어 볼 수도 있음
        - load.wav(), play()함수 이용
- 받아오는 파일이 압축파일일시 unzip() 함수 이용
    - zipfile : 압축파일이름 지정

    - 두번째 인수 : 압축 해제할 파일 이름 지정
- 일시적으로만 사용하고자 할 경우 : tempfile()로 임시파을 생성하고 unlink()이용 임시파일 삭제
- 테이블 형식의 정형화된 텍스트 파일이 아닌 불규칙한 구조일 경우 : scan() 함수 이용
    - 웹사이트상의 데이터 파일을 직접 읽어올 수 있음
        - 파일이름 대신 URL을 입력
        - skip 인수 : 설명/헤더 부분 몇라인 건너뛸지
        - nlines 인수 : 데이터가 몇라인에 걸쳐 있는지
        - what 인수 : 리스트형식으로 변수이름 / 데이터 유형 지정
    - 리스트 형식으로 저장됨, 데이터프레임으로(as.data.frame()) 변화하거나, 요약 통계량 계산(summary()) 가능
- 실제 웹페이지 : 다양한 HTML 태그 등으로 이루어져 이를 제거하는 작업 필요
    - 정규표현식을 이용하는 방법 : HTML 태그 패턴<...>을 정규표현식으로 표현, 제거
        - pattern <- "<.*?>" gsub(pattern,"",html)
        - 정교한 정규표현식을 만드는 것은 어려운 작업
    - XPath를 이용하는 방법 : XML 문서의 특정 부분에 접근하기 위한 경로를 지정하는 표기법
        - HTML 또한 XML처럼 노드가 트리구조로 표현, 따라서 XPath이용 가능
        - 상대경로 or 절대경로 이용가능
        - 각 단계는 axis::nodetest[predicate] 형식을 가짐
            - ㅅ/는 직계자식 노드, //는 모든 자식노드 의미
            - axis : 노드를 찾기위한 검색 방향 (ex) child, descendant(아래방향), ancestor(윗방향) 등
            - nodetest : 선택대상이 되는 노드(찾고자 하는 노드) (ex) td(td태그), text()(모든 텍스트 노드), comment()(주석 노드), node()(모든 유형의 노드)
            - predicate : 선택된 노드 중 특정 노드를 선택하기 위한 조건 지정(함수, 논리연산 사용가능)
                - child::td : 자식노드중 td 찾기
                - child::td[position()=1] : 자식노드중 첫번째 td 찾기
                - child::text() : 자식노드중 텍스트 노드 찾기
                - child::p/child::td[position()=5] : p노드의 자식노드중 다섯번째 td노드 찾기
                - td[attribute::class="xyz"] : td노드의 속성 중 class 값이 "xyz"인 것 찾기 (=td[@class="xyz"])
                - td[attribute::class="xyz"][position()=5] : 자식 노드 중 class 값이 "xyz"인 td 노드 가운데 다섯번째 것 찾기 (=td[@class="xyz"][5])
        - RCurl 패키지 getURL() 함수 : 웹페이지의 HTML 문서를 하나의 문자 벡터로 추출
        - XML 패키지 htmlParse() 함수 : 문자열로 추출한 HTML 웹페이지를 HTML 트리 구조를 갖는 R객체로 변환
        - XML 패키지 xpathSApply() : XPath를 바탕으로 HTML 문서와 같은 XML 형식의 문서로부터 특정 조건을 충족하는 노드들 추출
            - doc 인수 : 트리구조의 R 객체
            - path 인수 : 찾을 태그
            - fun 인수 : 해당 함수를 지정된 조건에 대응되는 노드들에 차례대로 적용
        - XML 패키지 xmlValue() 함수 : 태그 내의 텍스트 추출, 리프노드로부터 값 추출
            - 모든 노드에 적용 : sapply 사용
        - XML 패키지 readHTMLTable() 함수 : 문서에 포함된 테이블 데이터 읽어오기

            - doc 인수 : HTML 트리 구조 객체 / 문자 벡터 / URL 지정
            - stringsAsFactors=FALSE : 팩터 변환 방지
            - which 인수 : 테이블의 위치 지정
            - 리스트 형식으로 추출, 데이터값들은 리스트의 각 원소에 데이터프레임 형식으로 저장
                - 실제 데이터가 담긴 테이블만 나온다는 보장이 없음
    - 스크립트, 스타일 등 다양한 쓸데없는 내용이 포함되어 있을 가능성이 있기 때문에, 순수 텍스트만을 추출해 전처리 과정을 거치면 텍스트분석에 바로 활용 가능
    - 테이블의 수가 너무 많으면, 특정 테이블을 찾는 작업이 어려워짐 -> 효과적으로 찾는법?
        - 각 테이블별로 데이터 프레임의 시작과 끝에 포함된 데이터를 웹사이트상의 실제 테이블 데이터와 비교(단순한 반복작업) => 테이블의 개수가 많아지면 오래걸리고 비효율적
        - for 함수를 이용해 반복문 작성 가능
- API가 제공되는 웹사이트
    - quantmod 패키지 : API를 통해 주식 종목에 대한 데이터 다운로드
        - getSymbols() 함수에 Symbols 인수에 종목코드 지정 -> 해당 종목의 주가 및 거래량 데이터 다운로드
    - WDI 패키지 : 세계은행이 제공하는 다양한 통계 데이터
    - wbstats : 세계은행이 제공하는 공개 DB에 접근할 수 있는 환경 제공
        - WDI와의 차이점 : WDI는 와이드포맷 형태의 데이터셋 생성, wbstats는 롱포맷 형태의 데이터셋 생성

출력

화면

- 기본적으로 R 콘솔에 출력
- print() 함수 : 명시적으로 출력하기 위해서 이용, 객체만 입력해도 알아서 내부적으로 호출됨
    - 어떤 객체든 **적합한 형태**로 출력
    - 한번에 하나의 객체만을 출력할 수 있음
    - digits 인수 : 출력할 자릿수 조정(유효자릿수)
- cat() 함수 : 여러 객체를 하나로 이어서 출력, 객체 간 공백으로 구분
    - 라인 마무리 위해서 이스케이프문자 \n 를 인수로
    - 다양한 이스케이프문자 사용가능
    - 벡터만 출력 가능, 복합구조는 불가능 -> 벡터로 변환해 사용
    - cat 호출 전 format함수와 digits 인수로 자리수 지정 가능
        - 벡터가 주어지면 벡터 전체의 자리수가 한꺼번에 지정

파일

- cat() 함수 : file 인수를 지정해 파일형태로 출력
    - file 인수 : 파일명 지정
        - 직접입력하는대신, file()함수로 연결객체를 생성해 그 객체를 지정가능
            - 코드를 단순화시키고, 유지보수가 용이해짐
            - append옵션 지정이 필요없음
            - 연결종료위해 close()함수 이용

    - sep 인수 : 데이터 항목을 구분할 문자
    - append=TRUE 옵션 : 기존의 파일에 데이터 추가 / 사용안할시 덮어쓰기
- sink() 함수 : print()함수와 연계해 처리결과를 파일로 보낼수있음
    - 파일명을 인수로 지정 : 이후의 print()함수들이 화면이 아닌 파일로 보내짐
    - 다끝나면 인수없이 sink() 함수 호출
- write.csv() 함수 : CSV형식의 출력, 행렬이나 데이터프레임 같은 테이블 형식의 데이터를 CSV형식의 ASCII파일로 만듬, 행 = 파일의 한 라인, 항목 = 쉼표로 구분
    - 열의 이름이 자동으로 포함, 포함시키지 않으려면 col.names = FALSE 옵션 지정
    - 행 이름이 필요하지 않다면 row.names = FALSE 옵션 지정
- write.table() : 쉼표로 구분된 형식이 아닌 다른 형식으로 데이터 파일 생성
    - write.csv에 비해 다양한 옵션 제공
    - sep 인수 : 데이터 항목을 구분하는 구분자 지정, 기본값 = " "
    - append = TRUE 옵션 : 기존 파일에 새로운 데이터 추가
    - fileEncoding 인수 : 데이터 인코딩 방식 지정
- save() 함수 : 객체를 파일로 저장
    - load() 함수로 파일로 저장된 객체를 메모리에 적재
    - 인수를 지정하지 않으면 이진 형식으로 파일 저장
    - 텍스트형식으로 저장하려면 ascii=TRUE 옵션 지정
    - dput(), dump() 함수는 텍스트 파일 형식으로 객체 저장

클립보드

- writeClipboard() 함수 : 벡터 데이터를 클립보드에 복사
    - 실행시 R콘솔에는 결과 출력 X, 클립보드에만 복사
    - 테이블 형식의 데이터에는 적용 못함
- write.table() 함수: 테이블 형식의 데이터를 클립보드에 복사
    - file = "clipboard"로 인수 지정
    - R콘솔에는 결과 출력 X, 클립보드에만 복사

파일 탐색

- list.files() 함수 : 현재 작업 디렉토리 내의 파일 목록 보기
    - recursive = TRUE 옵션 지정 : 하위 디렉토리에 잇는 파일 목록까지 모두 보기
    - 숨겨진 파일은 보여주지 않음
    - all.files = TRUE로 모든 파일들의 목록 검색 가능
    - pattern 인수 : 특정 패턴을 갖는 파일의 목록 보기
    - path 인수 : 특정 디렉토리 내의 파일 목록 검색
- list.dirs() 함수 : 하위 디렉토리의 목록 모두 보기
- file.create() 함수 : 파일 생성
- file.remove() 함수 : 파일 제거
- file.exists() 함수 : 특정 파일이 존재하는지 확인

데이터베이스

- 데이터베이스의 데이터를 텍스트 형식의 파일로 저장한 후 해당 텍스트 파일 읽어오기

    - 대용량일경우 DB와의 직접연결보다 속도가 빠름
    - 일회성 작업일 경우 최적의 방법
    - 수시로 보고서 작성 / 반복작업이 필요한 경우 매번 텍스트로 변환해야하기때문에 비효율적
- RODBC : ODBC를 이용하여 DB에 연결, 표준 인터페이스제공, RODBC 패키지 이용
- DBI : DB에서 제공하는 고유의 DB 드라이버를 이요하여 DB에 연결, DBI(DB Interface) 패키지 제공

RODBC

- 연결할 데이터베이스에 대한 ODBC 드라이버를 설치한 후 ODBC 연결환경을 설정
- install.packages() 함수로 RODBC 패키지 설치
- ODBC 드라이버를 설치하고, ODBC 관리자를 열어 데이터베이스를 등록
- 사용자 DSN에서 새 데이터 원본을 만들고, 이름 / DB 선택등의 과정을 끝낸 뒤 odbcConnect() 함수로 연결
    - dsn 인수 : DSN 이름 지정
    - uid / pwd 인수 : 사용자명 / 패스워드가 설정되어있는경우 지정
    - 이 객체는 보통 채널이라고 불림
- 연결된 데이터베이스에 대한 정보를 보려면 odbcGetInfo() 함수 이용
    - channel 인수 : 정보를 볼 채널 객체
- sqlTables() 함수 : DB내에 어떤 테이블들이 포함되어 있는지 보기
    - channel 인수 : 채널 지정
    - 테이블 정보를 데이터 프레임 형식으로 반환
- sqlColumns() 함수 : 테이블 내의 열에 대한 정보 제공
    - channel 인수 : 채널 지정
    - sqtable 인수 : 테이블 이름 지정
- sqlFetch() 함수 : DB 내의 특정 테이블 가져오기
    - channel 인수 : 채널 지정
    - sqtable 인수 : 테이블 이름 지정
    - 지정된 테이블의 내용을 데이터프레임 형식으로 반환
    - 이후 subset() 등의 함수를 이용해 간단하게 데이터를 다룰 수 있음
- sqlQuery() 함수 : 데이터 처리시 SQL 쿼리를 사용
    - channel 인수 : 채널 지정
    - query 인수 : SQL문 지정
    - 결과를 데이터프레임 형식으로 반환
    - 어떠한 SQL 문장도 실행 가능
    - 테이블이 매우 크거나 쿼리가 복잡한 경우 실행시간이 다소 오래 걸림, 일부를 먼저 확인한 후 나중에 나머지 결과를 보는것이 때때로 유용
        - max 인수 : 한번에 가져올 행의 개수를 지정
- sqlGetResults() 함수 : sql연산결과 중 나머지 결과를 가져옴
- odbcClose() 함수 : 데이터베이스와의 연결통로 닫기
    - channel 인수 : 채널 지정
    - odbcCloseAll() 함수 : 연결된 모든 채널 동시 종료
- sqlSave() : 데이터프레임의 데이터를 데이터베이스에 저장
    - sqlUpdate() : 테이블을 업데이트할때
    - channel 인수 : 채널 지정

    - dat 인수 : 데이터프레임 이름
    - tablename 인수 : 생성할 테이블 이름
    - rownames 인수 : 데이터프레임의 행 이름이 저장될 테이블의 열 이름 지정
    - addPK = TRUE : 행 이름에 대응되는 열이 기본키로 사용

DBI

- 하나의 특정 패키지가 아니며, DB 접근을 위한 프레임워크이면서 패키지 집합을 의미
- 데이터베이스별로 제공, MySQL -> RMySQL, SQLite -> RSQLite, Oracle -> ROracle, PostgreSQL -> RPostgreSQL
- install.packages("RSQLite")로 설치
- dbConnect() 함수 : 데이터베이스 연결
    - drv 인수 : 드라이버 객체가 필요 -> dbDriver("SQLite")로 생성, 바로 연결객체를 생성해도 됨(SQLite())
    - dbname 인수 : 데이터베이스 파일 지정
- dbListTables() 함수 : 데이터베이스 내의 테이블 확인
    - conn 인수 : 연결 객체 지정
    - 테이블 이름을 **문자 벡터**로 반환
- dbListFields() 함수 : 테이블 내의 열 이름 확인
    - conn 인수 : 연결 객체 지정
    - name : 테이블 이름
    - 열 이름의 벡터 반환
- dbGetQuery() 함수 : SQL 쿼리 결과를 데이터프레임 형식으로 반환
    - conn 인수 : 연결 객체 지정
    - statement 인수 : SQL문
    - 테이블이 크거나 쿼리가 복잡한 경우 쿼리를 수행하는 작업 / 결과를 가져오는 작업의 분리가 필요
        - dbSendQuery() 함수 : SQL 쿼리를 보냄
            - conn 인수 : 연결 객체 지정
            - statement 인수 : SQL문
        - fetch() 함수 : 결과를 가져옴
            - n 인수 : 출력할 최대 행의 개수 지정, 지정하지 않을시 전체 결과 출력
        - dbClearResult() 함수 : 쿼리 결과 지우기
- dbReadTable() 함수 : SQL문장과 함께 테이블 전체 읽어오기, 좀 더 간단하게
    - conn 인수 : 연결 객체 지정
    - name : 테이블 이름 지정
- dbDisconnect() 함수 : 데이터베이스 연결 종료
    - conn 인수 : 연결 객체 지정
- dbUnloadDriver() 함수 : 데이터베이스 드라이버 객체 메모리에서 제거
    - 드라이버 객체 지정
- dbWriteTable() 함수 : 데이터프레임을 DBI를 이용하여 데이터베이스의 테이블로 만들기
- dbExistTable() 함수 : 테이블이 존재하는지 확인
- dbRemoveTable() 함수 : 테이블을 삭제