

R과 redshift를 항해할 때 알아야할 상식사전(+협업)

데이터야놀자 2018 트랙1 | 11:40 ~

<https://mrchypark.github.io/redshift-with-r>

[pdf버전] [문의하기] [의견 및 오류 신고]

스타누르기는 콘텐츠 제작자를 춤추게 합니다.

박찬엽

2018년 10월 19일

목차

1. 발표자 소개
2. R과 함께 redshift 라는 DB를 향해해보자
 - RPostgres vs RPostgreSQL
 - redshift 에서 schema
3. 날짜/시간 자료형
 - timezone 문제
 - 날짜 더하기

목차

1. 발표자 소개
2. R과 함께 redshift 라는 DB를 향해해보자
 - RPostgres vs RPostgreSQL
 - redshift 에서 schema
3. 날짜/시간 자료형
 - timezone 문제
 - 날짜 더하기

🍃 협업을 위해

- writexl vs openxlsx
- googledrive
- sendgridr
- keyring

발표자 소개



박찬엽

- (현)코빗 재무팀 데이터 담당자
 - 재무DB 구축/관리 및 자동화
- (전)서울도시가스 선행연구팀 연구원
 - 챗봇 엔진 개발 및 서버 구축
- (전)2017년 패스트 캠퍼스 데이터 분석 R 강의
 - 데이터 분석을 위한 중급 R 프로그래밍
- N사 뉴스 크롤러 N2H4, D사 뉴스 크롤러 DNH4 관리자
 - ForkonLP 프로젝트
- **FACEBOOK**@mrchypark
- **GITHUB**@mrchypark

블로그를 운영하고 있습니다.

Mrchypark

Home

Posts

Materials

Support

About

RSS

Twitter

Facebook

Instagram

LinkedIn

GitHub

© 2018. All rights reserved.

Built with Hugo

Theme Blackburn

Mrchypark

log for you & me | R, Docker, Text analysis etc

[패키지 소개] NoSQL 데이터베이스를 다루는 nodbi

2018 jan 08, 00:00

[nosql](#) / [dbi](#) / [package](#)

한줄요약 이제 R로 MongoDB, Redis(server based), CouchDB, Elasticsearch, etcd를 다룰 수 있습니다. R에는 여러가지 데이터 베이스를 다루는 도구들이 있습니다. MS의 표준 SQL 드라이버인 ODBC를 사용

[Read more »](#)

[패키지 소개] 암호화폐 시세와 인덱스를 제공하는 ubci 패키지를 소개합니다

2018 jan 18, 00:00

[ubci](#) / [crypto](#) / [package](#)

세줄요약 암호화폐 관련 시세와 인덱스 정보를 제공하는 ubci 패키지를 공개함. ubci는 upbit에서 제공하는 데이터로 비영리 배포는 자유라고 해서 개발함. 데이


[Read more »](#)

[패키지 소개] 네이버 뉴스와 다음 뉴스의 댓글 가져오기

2018 jan 14, 01:52

[N2H4](#) / [DNH4](#) / [comment](#) / [forkonlp](#)




열심히 R 생태계에 기여하고 있습니다.



author:mrchypark

/

Pull requestsIssuesMarketplaceExplore





CreatedAssignedMentionedReview requests


author:mrchypark


🔗 41 Open ✓ 139 Closed


Visibility Organization Sort


 junhewk/RcppMeCab add function install mecab-ko & dic for windows
#5 opened 3 hours ago by mrchypark


 junhewk/RcppMeCab pos() function only return Exception: list() when format = "list"
#4 by mrchypark was closed 3 hours ago 1


 junhewk/ktm Do you have any plan to pos tagger change to RmecabKo or RcppMeCab?
#1 opened 6 hours ago by mrchypark


 johndharrison/seleniumPipes is this project maintain states now?
#22 opened 17 hours ago by mrchypark


 forkonlp/N2H4 cran 등록
#81 opened 4 days ago by mrchypark


 ColinFay/fryingpane Do you have any plan to display col information like str() function?
#1 opened 7 days ago by mrchypark

 r-lib/liteq how can I use this with thor instead of sqlite?
#21 by mrchypark was closed 21 days ago 1







 rdpeng/threadpool Do you have any plan to explain how to use this to parallels?
#3 opened 26 days ago by mrchypark

 yunho0130/awesome-blockchain-kor api 추가
#7 by mrchypark was merged on 25 Jun 2

 amrrs/coinmarketcapr [question]any plan to change from Rcurl to curl or httr?
#10 opened on 13 Jun by mrchypark

 forkonlp/N2H4 댓글 df 데이터의 저장 에러

github 활동 많이 해주세요!

Git Awards				GitHub username	Search	Top users by city	Top users by country	Top users worldwide	About
Top R GitHub developers in South korea									
R		South korea		Submit					
Username			Country rank		Stars				
	forkonlp		1		105 ★				
	mrchypark		2		102 ★				
	cardiomoon		3		80 ★				
	joelcarlson		4		51 ★				
	youngwoos		5		43 ★				
	keencosmos		6		25 ★				

forkonlp

- **N2H4** : 댓글 수집이 가능한 네이버 뉴스 크롤러
- **DNH4** : 댓글 수집이 가능한 다음 뉴스 크롤러
- **presidentSpeechKr** : 대통령 연설문
- **stdkor** : 표준국어대사전 텍스트 자료

개인

- **sendgridr** : 메일 전송 서비스 sendgrid api 패키지
- **ubci** : 업비트 거래소 인덱스&시세 데이터
- **tqk** : 한국 주식 데이터
- **krlift** : 한국 승강기 현황 데이터
- **krlandprice** : 한국 표준공시지가 데이터

잘 부탁드립니다.

코빗은 비트코인 거래소

 KORBIT

시세 ▾트레이딩 ▾더보기 ▾ | 로그인회원가입

국내 최초 가상화폐 거래소

회원가입

통화	현재가 (₩)	변동 (24시간)	통화	현재가 (₩)	변동 (24시간)
비트코인	7,354,000	0.46% ▲	질리카	38	2.56% ▼
리플	514	2.65% ▼	이더리움 클래식	10,870	1.27% ▼
이더리움	228,800	1.02% ▼	라이트코인	59,300	0.92% ▼
비트코인 캐시	491,000	2.29% ▼	비트코인 골드	32,000	0.00% ▲

24시간 거래량 (₩)7,289,240,700

분석용 DW로 redshift를 사용하고 있음



R의 dplyr 로 db 다루는게 된다고 한거 같은데?!

dplyr의 가독성을 포기할 수 없었음

```
tbl(conn, 'flights') %>%
  group_by(tailnum) %>%
  summarise(count=n(),
             mean_distance = mean(distance, na.rm = TRUE),
             total_distance = sum(distance, na.rm = TRUE)) %>%
  filter(!is.na(tailnum))
```

```
## # Source:   lazy query [?? x 4]
## # Database: sqlite 3.22.0 []
##   tailnum count mean_distance total_distance
##   <chr>    <int>      <dbl>         <dbl>
## 1 D942DN      4        854.           3418
## 2 N0EGMQ    371        676.          250866
## 3 N10156    153        758.          115966
## 4 N102UW     48        536.           25722
## 5 N103US     46        535.           24619
## 6 N104UW     47        535.           25157
## 7 N10575    289        520.          150194
## 8 N105UW     45        525.           23618
## 9 N107US     41        529.           21677
## 10 N108UW    60        534.           32070
## # ... with more rows
```

걱정하지 마 미래의 나야, 훗!



연결할 수 있는가?

R Database

driver list (DBI supported)

- SQLite
- MySQL(+MariaDB)
- bigquery
- Oracle
- ...

+noDBI supported

- MongoDB
- Redis
- CouchDB
- Elasticsearch
- etcd
- ...

driver list (DBI supported)

- SQLite
- MySQL(+MariaDB)
- bigquery
- Oracle
- ...

+noDBI supported

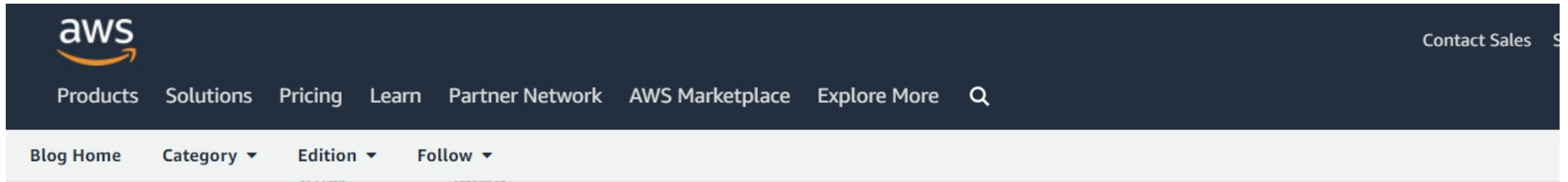
- MongoDB
- Redis
- CouchDB
- Elasticsearch
- etcd
- ...

+ sergeant (with Apache Drill)

- HBase
- MapR-DB
- HDFS
- MapR-FS
- Amazon S3
- ...

+ JDBC & ODBC

aws의 공식 문서는 jdbc를 안내함



Connecting R to Amazon Redshift with RJDBC

As soon as you have an R session and the data loaded to Amazon Redshift, you can connect them. The recommended connection method is using a client application or tool that executes SQL statements through the PostgreSQL ODBC or JDBC drivers.

In R, you can install the *RJDBC* package to load the JDBC driver and send SQL queries to Amazon Redshift. This requires a matching JDBC driver. Choose the **latest** JDBC driver provided by AWS ([Configure a JDBC Connection](#)).

This driver is based on the PostgreSQL JDBC driver but optimized for performance and memory management.

```
install.packages("RJDBC")
library(RJDBC)

# download Amazon Redshift JDBC driver
download.file('http://s3.amazonaws.com/redshift-downloads/drivers/RedshiftJDBC41-1.1.9.1009.jar', '

# connect to Amazon Redshift
```

jdbc 의 단점

1. java 설치

- 현재 버전의 rJava는 경로 문제가 해결되었습니다.

2. 숫자와 문자로만 자료형 제공

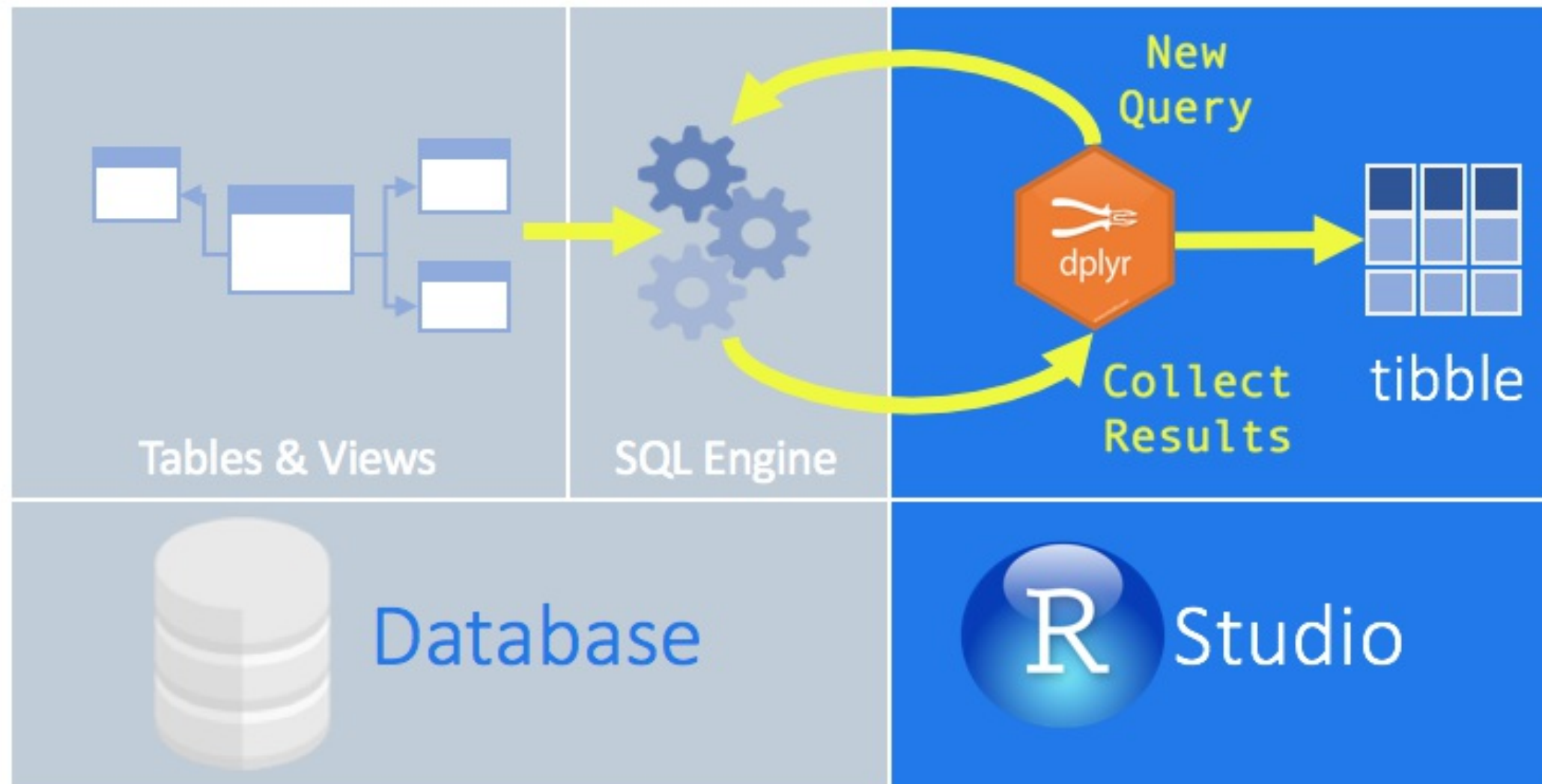
- 날짜, 논리형, 시간 등 모두 문자 자료형으로 제공

3. 정밀도가 떨어짐.

- 예를 들어 9223372036854775807를 9223372036854775808로 리턴해줌

DBI

Use dplyr to interact with the database



RPostgreSQL & RPostgres

RPostgreSQL

1. ssl 등 보안 연결 지원 미흡

RPostgreSQL

1. ssl 등 보안 연결 지원 미흡

회사에서 사용하기 어려움...

RPostgres

1. postgres 팀의 표준 라이브러리(libpq) 사용
2. DBI 패키지 관리 팀이 함께 지원 관리
3. 연결 풀 및 메모리 자동 관리

연결해 보자

```
library(DBI)
conn <- dbConnect(RPostgres::Postgres(), # 그대로 사용
                  host = host,            # redshift의 endpoint 주소
                  port = port,            # 사용하는 port redshift는 기본값이 5439
                  user = user,            # DB 계정(소위 ID)
                  password = password,    # 비밀번호
                  dbname = dbname,         # 필수 아님. redshift에 기본 db가 표시됨
                  sslmode='require'       # 암호 연결 요구
                  )
```

연결 성공! 했지만

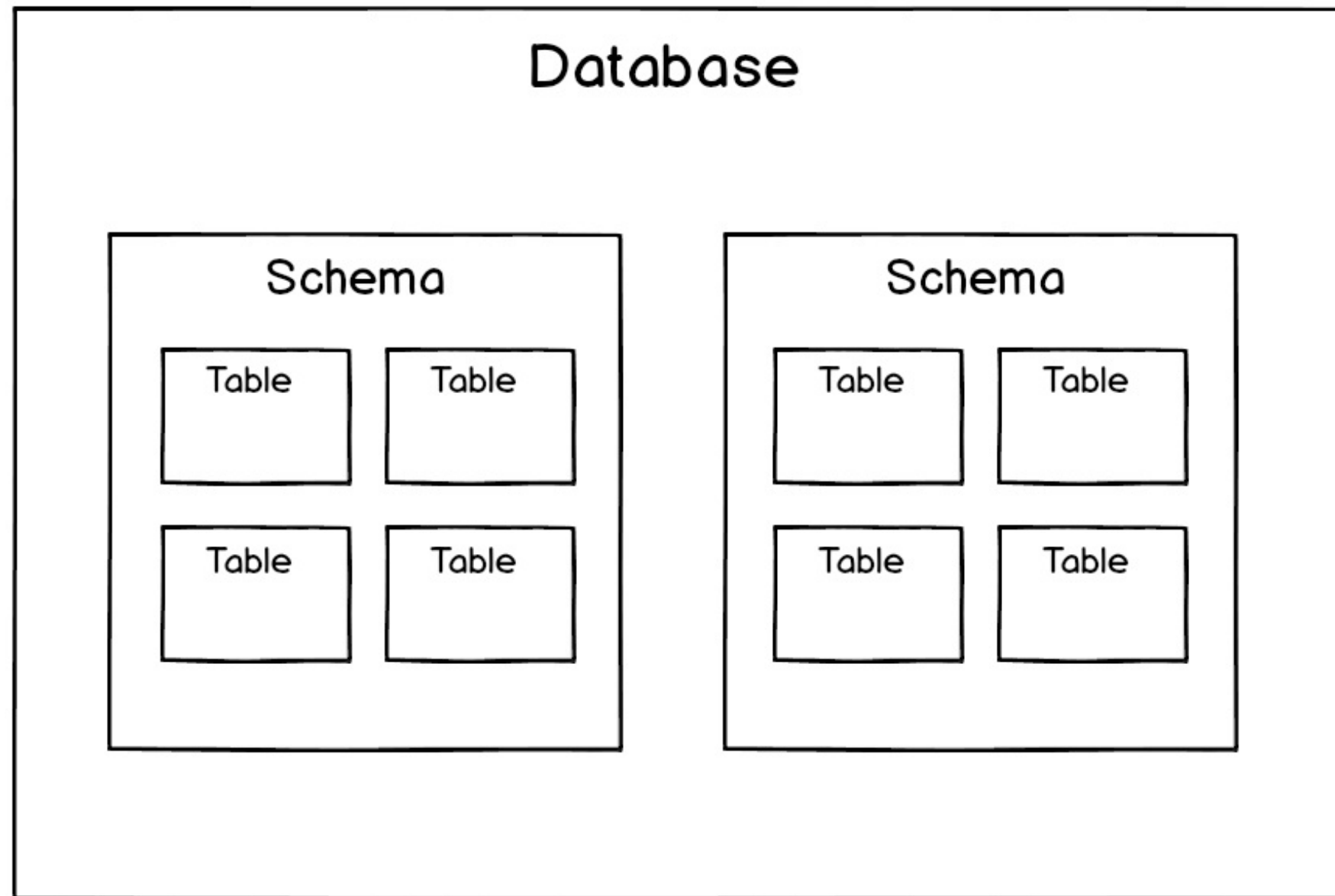
테이블이 없다?!

```
dbListTables(conn)
```

```
## character(0)
```

스키마를 통해 권한 관리

실제 스키마가 뭔지는 잘 모르겠지만 폴더 같은 역할을 수행(postgres)



스키마 내의 테이블 리스트를 가져오는 법

postgres에서 스키마 내의 테이블을 보는 방법을 찾아보니 쿼리문이 있음

```
dbGetQuery("SELECT * FROM information_schema.tables WHERE table_schema = 'public'")
```

스키마 내의 테이블 리스트를 가져오는 법

postgres에서 스키마 내의 테이블을 보는 방법을 찾아보니 쿼리문이 있음

```
dbGetQuery("SELECT * FROM information_schema.tables WHERE table_schema = 'public'")
```

전부 R로 하려고 고집 부리지 않는다

redshift에서 스키마 내의 테이블 불러오기

테이블 명으로 dplyr의 tbl 형으로 찾을 수 없다고 출력

```
tbl(con, "table_name")
```

```
## Error in result_create(conn@ptr, statement) :  
## Failed to prepare query: ERROR: relation "table_name" does not exist
```

redshift에서 스키마 내의 테이블 불러오기

테이블 명으로 dplyr의 tbl 형으로 찾을 수 없다고 출력

```
tbl(con, "table_name")
```

```
## Error in result_create(conn@ptr, statement) :  
## Failed to prepare query: ERROR: relation "table_name" does not exist
```

schema.table_name 형태로 스키마를 선언해줘도 같은 문제 발생

```
tbl(con, "schema.table_name")
```

```
## Error in result_create(conn@ptr, statement) :  
## Failed to prepare query: ERROR: relation "schema.table_name" does not exist
```

schema.table_name로 테이블 지정

sql () 함수를 이용해서 쿼리한 테이블을 연결해줘야 함.

```
tbl(con, sql("select * from schema.table_name"))
```

dbplyr 패키지

~~다른 좋은 함수가 많겠지만!~~ in_schema 명령어 지원

```
if (!requireNamespace("dbplyr")) install.packages("dbplyr")  
library(dbplyr)
```

```
##  
## Attaching package: 'dbplyr'  
  
## The following objects are masked from 'package:dplyr':  
##  
##     ident, sql
```

```
tbl(con, in_schema("schema", "table_name"))
```

dplyr에서 유용한 함수

collect()

collect()는 DB에 전달하는 명령의 최종 결과를 R 객체로 가져오는 역할을 수행

```
tb_flights %>%  
  group_by(carrier) %>%  
  summarise(count = n())
```

```
# Source:   lazy query [?? x 2]  
# Database: sqlite 3.22.0 [:memory:]  
  carrier count  
  <chr>    <int>  
1  9E      18460  
2  AA      32729  
3  AS        714  
4  B6     54635  
5  DL     48110  
6  EV     54173  
7  F9        685  
8  FL     3260  
9  HA        342
```

```
tb_flights %>%  
  group_by(carrier) %>%  
  summarise(count = n()) %>%  
  collect()
```

```
## # A tibble: 16 x 2  
##   carrier count  
##   <chr>    <int>  
## 1  9E      18460  
## 2  AA      32729  
## 3  AS        714  
## 4  B6     54635  
## 5  DL     48110  
## 6  EV     54173  
## 7  F9        685  
## 8  FL     3260  
## 9  HA        342
```

show_query()

show_query()는 dplyr로 구성된 함수의 연결이 query문으로 어떻게 변환되는지를 보여줌

```
copy_to(conn, planes, name = 'planes', temporary = FALSE)
tbl(conn, 'planes_distance') %>%
  inner_join(tbl(conn, 'planes'), by='tailnum') %>%
  arrange(desc(total_distance)) %>%
  select(total_distance, manufacturer, model) %>%
  show_query()
```

```
<SQL>
SELECT `total_distance`, `manufacturer`, `model`
FROM (SELECT *
FROM (SELECT `TBL_LEFT`.`tailnum` AS `tailnum`, `TBL_LEFT`.`count` AS `count`, `TBL_LEFT`.`mean`
FROM `planes_distance` AS `TBL_LEFT`
INNER JOIN `planes` AS `TBL_RIGHT`
ON (`TBL_LEFT`.`tailnum` = `TBL_RIGHT`.`tailnum`)
)
ORDER BY `total_distance` DESC)
```

< >

이제 dplyr을 잘 사용하면 되는건가?

거래소는 24시간이 모자라

박찬엽



- (현)코빗 재무팀 데이터 담당자
- (전)서울도시가스 선행연구팀 연구원
 - 챗봇 엔진 개발 및 서버 구축
- (전)2017년 패스트 캠퍼스 데이터 분석 R 강의
 - 데이터 분석을 위한 중급 R 프로그래밍
- N사 뉴스 크롤러 N2H4, D사 뉴스 크롤러 DNH4 관리자
 - ForkonLP 프로젝트
- FACEBOOK@mrchypark
- GITHUB@mrchypark

재무팀의 시간

일단위(**daily**) 데이터 처리

시간이 매우 중요하다.

1. 여러 테이블들의 관계에서 기준 시간을 정해야 함.
 - 수많은 created_at과 updated_at 사이에서 찾아야...
2. timezone 확인
 - 외국인 등 이유로 UTC(협정 세계시) 사용
3. 거래 취소 등이 나중에 발생하기도 함
 - 18일에 확인한 17일 값과 19일에 확인한 17일 값이 다를 수 있음

날짜/시간 자료형이 어려워서

year, month, day 같이 컬럼별로 쪼개 두기도 함

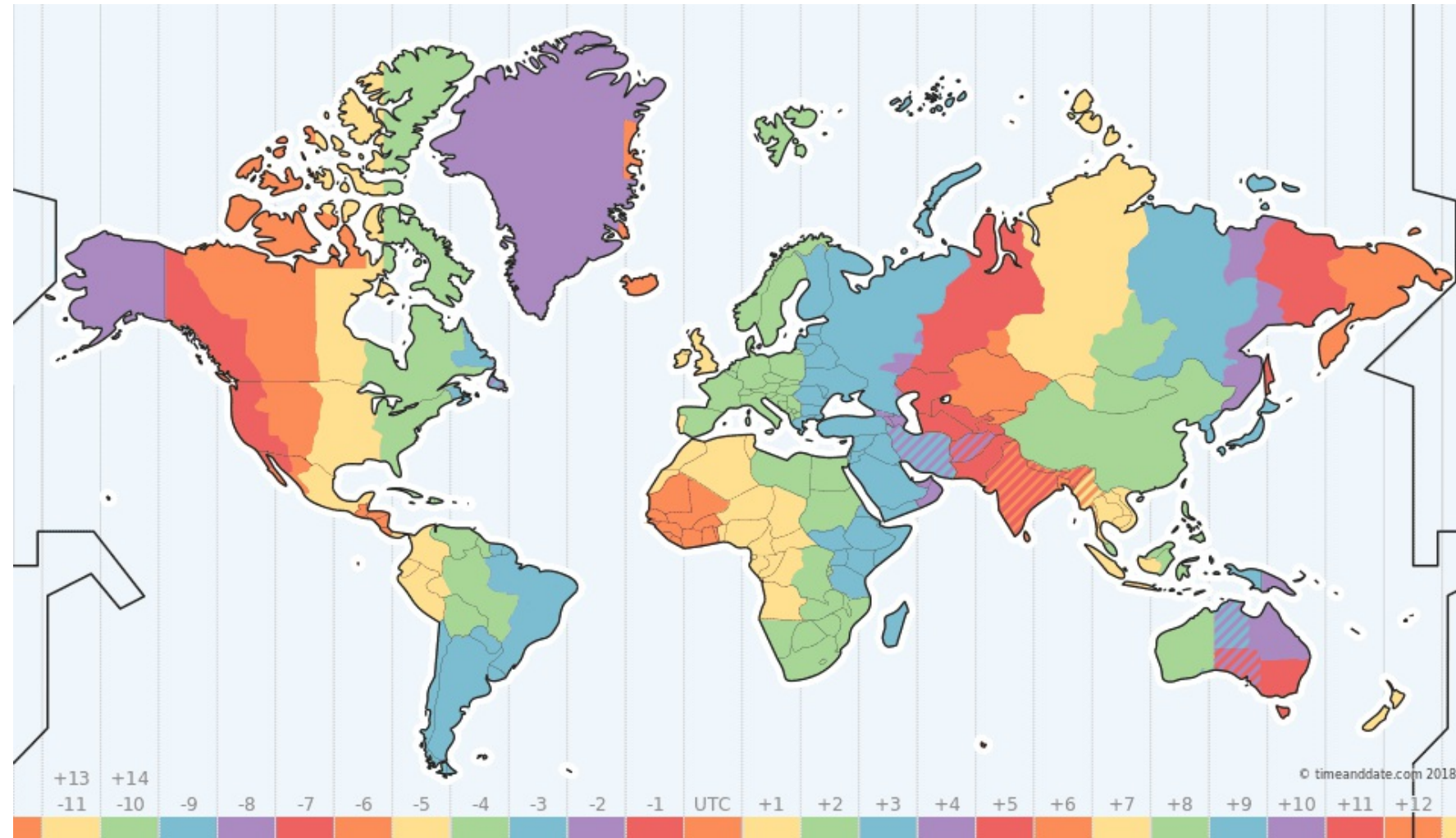
```
library(nycflights13)
flights
```

```
## # A tibble: 336,776 x 19
##   year month   day dep_time sched_dep_time dep_delay arr_time
##   <int> <int> <int>   <int>         <int>      <dbl>    <int>
## 1  2013     1     1     517           515         2      830
## 2  2013     1     1     533           529         4      850
## 3  2013     1     1     542           540         2      923
## 4  2013     1     1     544           545        -1     1004
## 5  2013     1     1     554           600        -6      812
## 6  2013     1     1     554           558        -4      740
## 7  2013     1     1     555           600        -5      913
## 8  2013     1     1     557           600        -3      709
## 9  2013     1     1     557           600        -3      838
## 10 2013     1     1     558           600        -2      753
## # ... with 336,766 more rows, and 12 more variables: sched_arr_time <int>,
## #   arr_delay <dbl>, carrier <chr>, flight <int>, tailnum <chr>,
## #   origin <chr>, dest <chr>, air_time <dbl>, distance <dbl>, hour <dbl>,
## #   minute <dbl>, time_hour <dtm>
```

날짜/시간 자료형

잘 만들어진 시스템을 잘 사용해보자

timezone



한국은?

한국은?

Asia/Seoul(KST)

한국은?

Asia/Seoul(KST)

UTC(협정 세계시) + 9시간

Timezones in base R

UTC 그런거 없음

```
x <- "2018-01-01 06:00:00 UTC"  
as.POSIXct(x)
```

```
## [1] "2018-01-01 06:00:00 KST"
```

```
as.POSIXct(x, tz = "Asia/Seoul")
```

```
## [1] "2018-01-01 06:00:00 KST"
```

```
Sys.timezone()
```

```
## [1] "Asia/Seoul"
```

```
Sys.time()
```

```
## [1] "2018-10-19 08:11:31 KST"
```

redshift는 UTC가 기본

postgres로 실습 환경 구성

```
library(DBI)
library(dplyr)
library(tibble)
library(lubridate)

con <- dbConnect(RPostgres::Postgres(),
                 host = '192.168.99.100',
                 port = 5432,
                 user = 'postgres',
                 password = '1q2w3e')

dtdata <- as.character(Sys.time()+sample(1:1000000,10))
dtdata <- tibble(time = dtdata) %>%
  mutate(time = as_datetime(time)) # lubridate::as_datetime() UTC 시간을 만들어 줌

copy_to(con, dtdata, overwrite =T)
dbListTables(con)
```

```
## [1] "dtdata"
```

as.Date 함수의 연습

```
tbl(con, "dtdata") %>%
  mutate(add = time %+ interval% '9 hours') %>%
  collect() %>%
  mutate(date = as.Date(time),
         datez = as.Date(time, tz = "Asia/Seoul"),
         add9 = as.Date(add),
         add9z = as.Date(add, tz = "Asia/Seoul"))
```

```
## # A tibble: 10 x 6
##   time                add                date                datez                add9
##   <dtm>              <dtm>              <date>              <date>              <date>
## 1 2018-10-25 05:37:35 2018-10-25 14:37:35 2018-10-24 2018-10-25 2018-10-25
## 2 2018-10-26 20:59:24 2018-10-27 05:59:24 2018-10-26 2018-10-26 2018-10-26
## 3 2018-10-20 15:53:53 2018-10-21 00:53:53 2018-10-20 2018-10-20 2018-10-20
## 4 2018-10-26 11:35:50 2018-10-26 20:35:50 2018-10-26 2018-10-26 2018-10-26
## 5 2018-10-21 11:20:45 2018-10-21 20:20:45 2018-10-21 2018-10-21 2018-10-21
## 6 2018-10-22 09:01:31 2018-10-22 18:01:31 2018-10-22 2018-10-22 2018-10-22
## 7 2018-10-21 18:22:48 2018-10-22 03:22:48 2018-10-21 2018-10-21 2018-10-21
## 8 2018-10-28 17:02:17 2018-10-29 02:02:17 2018-10-28 2018-10-28 2018-10-28
## 9 2018-10-23 21:44:18 2018-10-24 06:44:18 2018-10-23 2018-10-23 2018-10-23
## 10 2018-10-23 21:55:16 2018-10-24 06:55:16 2018-10-23 2018-10-23 2018-10-23
## # ... with 1 more variable: add9z <date>
```

헛갈리기 딱 좋음

1. `as.Date()` 함수가 알아서 UTC(?)로 되돌려 놓음.
 - 기본 R에서의 동작과 다름
2. `time %+ interval% '9 hours'` 같은 DB쪽 문법을 사용해야 함.
 - `time + hour(9)` 같은거 알아서 바꿔주지 않음 ㅜㅜ
 - `dplyr::mutate`의 소스가 `db_con` 일 경우 글자 수준으로 sql 문에 전달
 - `redshift`는 `convert_timezone()` 함수를 지원함
 - `convert_timezone(time, "kst")` 같은 문법이 가능함

느낀 점

- 역시 공짜 점심은 없다
- 어느 정도 각 DB 특성에 맞는 지식을 알고 있어야
- 날짜/시간 자료형은 어디서나 문제니 꼭 UTC 기준인지 확인 필요
- timezone 단위로 단위로 처리하는 것이 추상화된 해결책
- 다행히 인코딩 문제는 익숙한데... UTF-8을 확인하는 것이 중요 포인트

협업을 위한 패키지들

csv 대신 엑셀로 저장하기 - writexl

```
library(openxlsx)
library(writexl)
```

```
library(microbenchmark)
library(nycflights13)
microbenchmark(
  writexl = writexl::write_xlsx(flights, tempfile()),
  openxlsx = openxlsx::write.xlsx(flights, tempfile()),
  times = 3
)
```

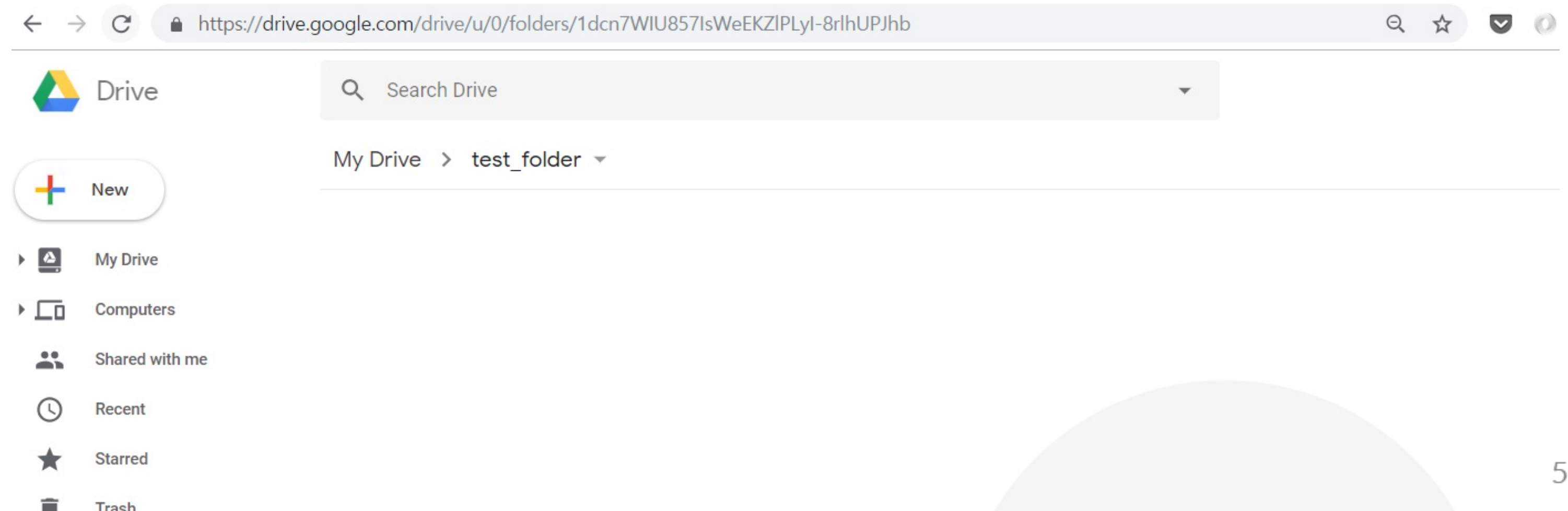
```
## Unit: seconds
##      expr      min       lq      mean     median        uq      max  neval
##  writexl  8.884712  8.904431  9.103419  8.965643  9.041565  9.720743     5
## openxlsx 17.166818 18.072527 19.171003 18.669805 18.756661 23.189206     5
```

```
writexl::write_xlsx(flights, tmp1 <- tempfile())
file.info(tmp1)$size
## 29157282
openxlsx::write.xlsx(flights, tmp2 <- tempfile())
file.info(tmp2)$size
## 35962067
```


구글 드라이브

엑셀 파일 업로드

```
library(googledrive)
drive_upload(media = "what_you_want_to_upload_file",
             name = "name_of_upload_file",
             path = as_id("where_to_upload"))
```



메일 보내기

rmd로 작성한 보고서 전송

구글에서 이미지를 제거해서 임시로 html을 첨부하는 형태로 우회중

```
mail() %>%  
  from("your@mail.com") %>%  
  to("where@togo.com") %>%  
  subject("mail title") %>%  
  content("mail body") %>%  
  attachments("what_you_want_to_upload_file") %>%  
  send()
```

보안을 위해 keyring

windows의 자격증명관리자, mac의 Keychain 을 이용해 비밀 번호 등을 암호화 저장후 활용

```
keyring::key_set("key-name-to-use")
con <- dbConnect(RPostgres::Postgres(),
  host = '192.168.99.100',
  port = 5432,
  user = 'postgres',
  password = keyring::key_get("key-name-to-use"))
```

깃헙등에 잘못 올릴 일이 없음

Q & A

Q & A

감사합니다