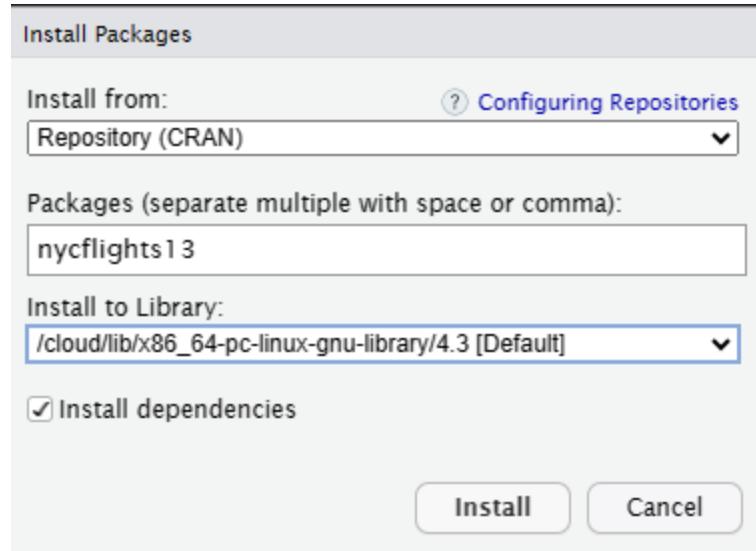


Project 3: Data Transformation

▼ Instruction for Homework

Assignment_Data_Trasform.R

- Load Package: `nycflights13`



```

> install.packages("nycflights13")
Installing package into ‘/cloud/lib/x86_64-pc-linux-gnu-library/4.3’
(as ‘lib’ is unspecified)
trying URL 'http://rspm/default/_linux/_focal/latest/src/contrib/nycflights13_1.0.2.tar.gz'
Content type 'application/x-gzip' length 4506293 bytes (4.3 MB)
=====
downloaded 4.3 MB

* installing *binary* package ‘nycflights13’ ...
* DONE (nycflights13)

The downloaded source packages are in
  ‘/tmp/RtmpnmgEBd/downloaded_packages’
> |

```

- Load Library 2 တဲ့
 - nycflights13
 - tidyverse
- Create Question 5 Questions by using select, filter, arrange, mutate, summarise, count

```

## homework

## transform nycflights13

library(nycflights13)
library(tidyverse) # dplyr

## ask 5 questions about this data set

## load data to memory
data("flights")
## Abbr. Airline
data("airlines")
## Know about each attribute of data set
?flights

## select filter arrange mutate summarise count

## Example Q. most flight carrier in sep 2013

```

```
flights %>%  
  filter (month == 9 , year == 2013) %>%  
  count(carrier) %>%  
  arrange(desc(n)) %>%  
  head(5)
```

```
> library(nycflights13)  
> library(tidyverse) # dplyr  
>  
> ## ask 5 questions about this data set  
>  
> ## load data to memory  
> data("flights")  
> ## Know about each attribute of data set  
> ?flights  
>  
>  
> ## select filter arrange mutate summarise count  
>  
> ## Example Q. most flight carrier in sep 2013  
> flights %>%  
+   filter (month == 9 , year == 2013) %>%  
+   count(carrier) %>%  
+   arrange(desc(n)) %>%  
+   head(5)  
# A tibble: 5 × 2  
  carrier     n  
  <chr>    <int>  
1 EV        4725  
2 UA        4694  
3 B6        4291  
4 DL        3883  
5 AA        2614
```



Detail of Data Set - flights by using command in console : `?flights`

flights {nycflights13} R Documentation
Flights data
Description
On-time data for all flights that departed NYC (i.e. JFK, LGA

Usage
flights

Format

Data frame with columns

`year, month, day`

Date of departure.

`dep_time, arr_time`

Actual departure and arrival times (format HHMM or HMM), loca

`sched_dep_time, sched_arr_time`

Scheduled departure and arrival times (format HHMM or HMM), l

`dep_delay, arr_delay`

Departure and arrival delays, in minutes. Negative times repr

`carrier`

Two letter carrier abbreviation. See `airlines` to get name.

`flight`

Flight number.

`tailnum`

Plane tail number. See `planes` for additional metadata.

`origin, dest`

Origin and destination. See `airports` for additional metadata.

`air_time`

Amount of time spent in the air, in minutes.

`distance`

Distance between airports, in miles.

`hour, minute`

Time of scheduled departure broken into hour and minutes.

time_hour

Scheduled date and hour of the flight as a POSIXct date. Along

Source

RITA, Bureau of transportation statistics, <https://www.transportation.gov>

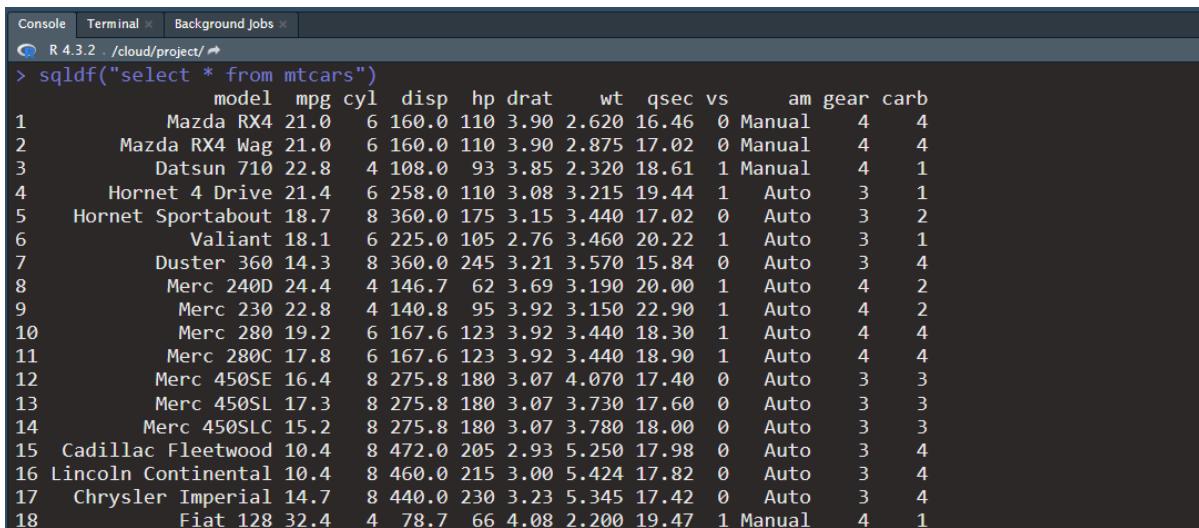
Tip : ถ้าใช้ SQL Statement ต้อง load `Package : sqldf` และใช้ `library : sqldf`

```
### Package : sqldf
install.packages("sqldf")
library(sqldf)

sqldf("select * from mtcars")

sqldf("select mpg, hp from mtcars
      where hp >= 200")

sqldf("select avg(hp),max(hp),count(*) from mtcars")
```



The screenshot shows the RStudio interface with the 'Console' tab selected. The command `> sqldf("select * from mtcars")` is entered and its output is displayed. The output is a data frame with 32 rows and 11 columns, representing the mtcars dataset.

	model	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
1	Mazda RX4	21.0	6	160.0	110	3.90	2.620	16.46	0	Manual	4	4
2	Mazda RX4 Wag	21.0	6	160.0	110	3.90	2.875	17.02	0	Manual	4	4
3	Datsun 710	22.8	4	108.0	93	3.85	2.320	18.61	1	Manual	4	1
4	Hornet 4 Drive	21.4	6	258.0	110	3.08	3.215	19.44	1	Auto	3	1
5	Hornet Sportabout	18.7	8	360.0	175	3.15	3.440	17.02	0	Auto	3	2
6	Valiant	18.1	6	225.0	105	2.76	3.460	20.22	1	Auto	3	1
7	Duster 360	14.3	8	360.0	245	3.21	3.570	15.84	0	Auto	3	4
8	Merc 240D	24.4	4	146.7	62	3.69	3.190	20.00	1	Auto	4	2
9	Merc 230	22.8	4	140.8	95	3.92	3.150	22.90	1	Auto	4	2
10	Merc 280	19.2	6	167.6	123	3.92	3.440	18.30	1	Auto	4	4
11	Merc 280C	17.8	6	167.6	123	3.92	3.440	18.90	1	Auto	4	4
12	Merc 450SE	16.4	8	275.8	180	3.07	4.070	17.40	0	Auto	3	3
13	Merc 450SL	17.3	8	275.8	180	3.07	3.730	17.60	0	Auto	3	3
14	Merc 450SLC	15.2	8	275.8	180	3.07	3.780	18.00	0	Auto	3	3
15	Cadillac Fleetwood	10.4	8	472.0	205	2.93	5.250	17.98	0	Auto	3	4
16	Lincoln Continental	10.4	8	460.0	215	3.00	5.424	17.82	0	Auto	3	4
17	Chrysler Imperial	14.7	8	440.0	230	3.23	5.345	17.42	0	Auto	3	4
18	Fiat 128	32.4	4	78.7	66	4.08	2.200	19.47	1	Manual	4	1

```

Console Terminal × Background Jobs ×
R 4.3.2 . /cloud/project/ ↗
> sqldf("select mpg, hp from mtcars
+           where hp >= 200")
   mpg  hp
1 14.3 245
2 10.4 205
3 10.4 215
4 14.7 230
5 13.3 245
6 15.8 264
7 15.0 335
>
> sqldf("select avg(hp),max(hp),count(*) from mtcars")
   avg(hp) max(hp) count(*)
1 146.6875     335         32
> |

```

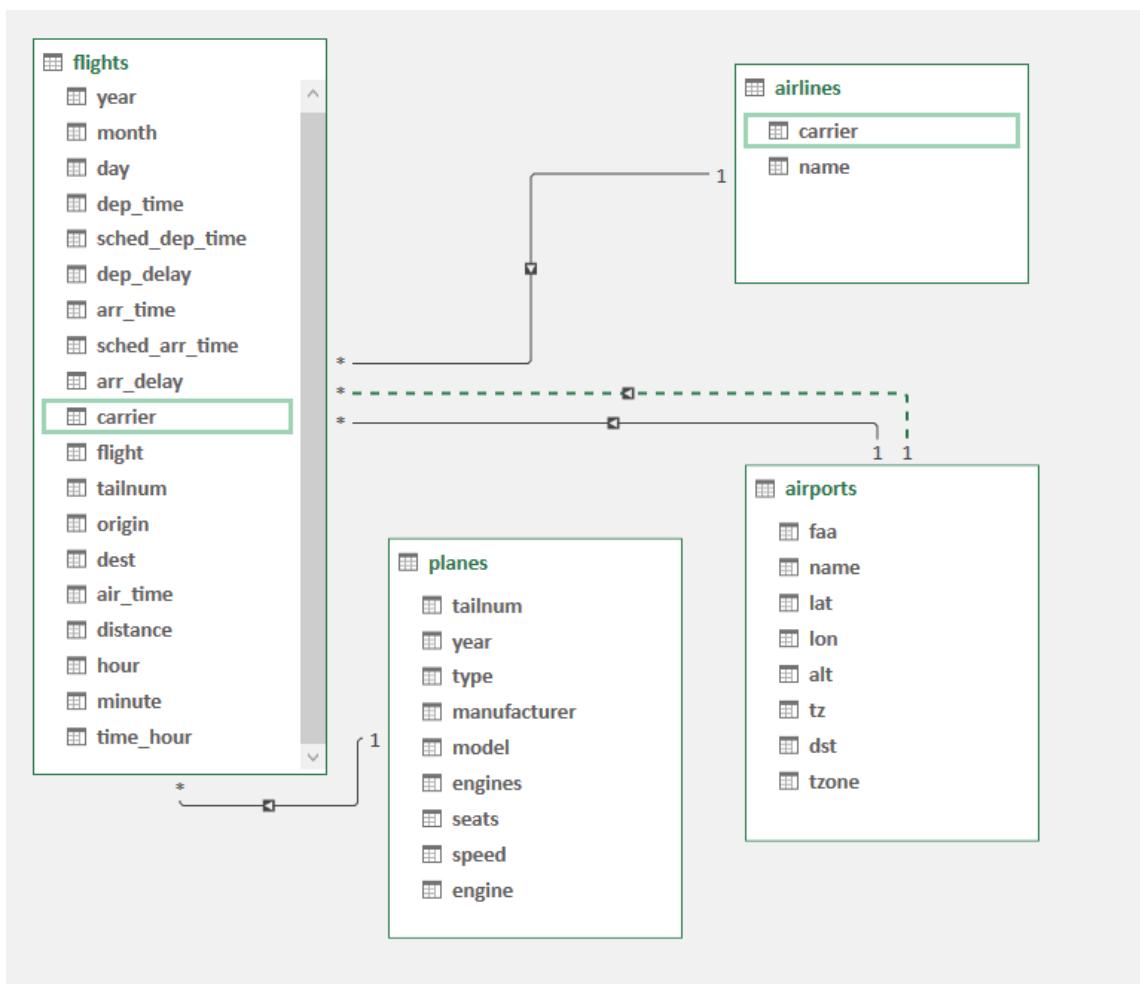
▼ Assign Homework (HW1)

- HW 1 - Create SQL from a data set of `nycflights13`
 - 5 Questions by using select, filter, summarise, mutate, arrange, count()



About Data Set

- Select 4 Data Sets that are
 - Flights Data (`flights`): on-time data for all flights that departed NYC (i.e. JFK, LGA or EWR) in 2013
 - Airline names (`airlines`): Look up airline names from their carrier codes.
 - Airport metadata (`airports`): Useful metadata about airports.
 - Plane metadata(`planes`): Plane metadata for all plane tail numbers found in the FAA aircraft registry. American Airway (AA) and Envoy Air (MQ) report fleet numbers rather than tail numbers so can't be matched.



Example Data : Project_Data_Transform.csv



Preparing Environment - load Library

```
## homework

## transform nycflights13

library(nycflights13)
library(tidyverse) # dplyr
```

```
## Ask 5 questions about this data set

## Load data to memory
data("flights")
## Know about each attribute of the data set
?flights
```

Example: Assignment_Data_Trasform_R

Assignment1_Q1_5.R



Q1: In month 9, which manufacturer uses airplanes the most? list the Top 10 companies.

```
flights %>%
  filter (month == 9) %>%
  inner_join(planes,by = "tailnum") %>%
  group_by(month,manufacturer) %>%
  summarise(cnt = n()) %>%
  arrange(desc(cnt)) %>%
  head(10)
```

Result Set

```

> ## select filter arrange mutate summarise count
>
> ## Q1 : In month = 9, which manufacturer uses airplanes the most? Top 10 companies.
>
> flights %>%
+   filter (month == 9) %>%
+   inner_join(planes,by = "tailnum") %>%
+   group_by(month,manufacturer) %>%
+   summarise(cnt = n()) %>%
+   arrange(desc(cnt)) %>%
+   head(10)
`summarise()` has grouped output by 'month'. You can override using the `.`groups` argument.
# A tibble: 10 × 3
# Groups:   month [1]
  month manufacturer          cnt
  <int> <chr>            <int>
1     9 BOEING           6425
2     9 EMBRAER          5343
3     9 AIRBUS            3872
4     9 AIRBUS INDUSTRIE 3516
5     9 BOMBARDIER INC  2786
6     9 MCDONNELL DOUGLAS AIRCRAFT CO 657
7     9 MCDONNELL DOUGLAS        344
8     9 CANADAIR          148
9     9 MCDONNELL DOUGLAS CORPORATION    84
10    9 CESSNA             73
>
```



Q2: Summarize the number of flights in each airline for month 9.

```

flights %>%
  filter (month == 9) %>%
  inner_join(airlines,by = "carrier") %>%
  group_by(carrier,name) %>%
  summarise(cnt = n()) %>%
  arrange(desc(cnt))
```

Result Set

```

> ## Q2 : summarize number of flight in each airline of month 9
> flights %>%
+   filter (month == 9) %>%
+   inner_join(airlines,by = "carrier") %>%
+   group_by(carrier,name) %>%
+   summarise(cnt = n()) %>%
+   arrange(desc(cnt))
`summarise()` has grouped output by 'carrier'. You can override using the `.`groups` argument.
# A tibble: 16 × 3
# Groups:   carrier [16]
  carrier name          cnt
  <chr>   <chr>      <int>
1 EV      ExpressJet Airlines Inc. 4725
2 UA      United Air Lines Inc. 4694
3 B6      JetBlue Airways 4291
4 DL      Delta Air Lines Inc. 3883
5 AA      American Airlines Inc. 2614
6 MQ      Envoy Air 2206
7 US      US Airways Inc. 1698
8 9E      Endeavor Air Inc. 1540
9 WN      Southwest Airlines Co. 1010
10 VX     Virgin America 453
11 FL     AirTran Airways Corporation 255
12 AS     Alaska Airlines Inc. 60
13 F9     Frontier Airlines Inc. 58
14 YV     Mesa Airlines Inc. 42
15 HA     Hawaiian Airlines Inc. 25
16 OO     SkyWest Airlines Inc. 20
>

```



Q3: Identify the top 10 airplane types and manufacturers that have the maximum total distance and the total amount of time spent in the air.

```

flights %>%
  filter (year == 2013) %>%
  inner_join(planes,by = "tailnum") %>%
  group_by (manufacturer, type, engine) %>%
  summarise(Total_distance    = sum(distance,na.rm = TRUE),
            Total_AirTime_min = sum(air_time,na.rm= TRUE),
            Total_AirTime_hr  = round(sum(air_time,na.rm= TRUE),
            cnt = n()) %>%
  arrange(desc(Total_distance)) %>%
  head(10)

```



```

> ## Q3 : Identify the top 10 types of airplanes and manufacturers that have the maximum total distance and the total amount of time spent in the air
> flights %>%
+   filter(year == 2013) %>%
+   inner_join(planes, by = "tailnum") %>%
+   group_by(manufacturer, type, engine) %>%
+   summarise(Total_distance = sum(distance, na.rm = TRUE),
+             Total_AirTime_min = sum(air_time, na.rm = TRUE),
+             Total_AirTime_hr = round(sum(air_time, na.rm = TRUE)/60, 2),
+             cnt = n()) %>%
+   arrange(desc(Total_distance)) %>%
+   head(10)
`summarise()` has grouped output by 'manufacturer', 'type'. You can override using the
`.groups` argument.
# A tibble: 10 × 7
# Groups: manufacturer, type [8]
  manufacturer     type   engine Total_distance Total_AirTime_min Total_AirTime_hr   cnt
  <chr>           <chr> <chr>        <dbl>            <dbl>          <dbl> <int>
1 BOEING          Fixe... Turbo...    81196481       11198044      186634.  55292
2 AIRBUS          Fixe... Turbo...    66963735       9142196       152370.  46705
3 BOEING          Fixe... Turbo...    48583727       6569012       109484.  27620
4 EMBRAER         Fixe... Turbo...    34462616       5370694       89512.   65794
5 AIRBUS INDUSTRIE Fixe... Turbo...    29703497       4277392       71290.   30379
6 BOMBARDIER INC Fixe... Turbo...    14990924       2341640       39027.   28272
7 AIRBUS INDUSTRIE Fixe... Turbo...    10414105       1509103       25152.   10512
8 MCDONNELL DOUGLAS AI... Fixe... Turbo...    8156412        1172515      19542.   8932
9 MCDONNELL DOUGLAS Fixe... Turbo...    3751769        538464       8974.    3873
10 MCDONNELL DOUGLAS CO... Fixe... Turbo...   1141928        165057       2751.    1259

```



Q4: Create a New Column speed_mph and summarize the top 10 of the mean speed for each airline company.

```

flights %>%
  filter (year == 2013) %>%
  mutate (speed_mph = distance / (hour + (minute/60))) %>%
  inner_join(airlines, by = "carrier") %>%
  group_by(name, origin, dest) %>%
  summarise(mean_speed = mean(speed_mph)) %>%
  arrange(name, desc(mean_speed)) %>%
  head(10)

```

Result Set

```

> ## Q4 : Create New Column speed_mph and Summarize mean speed for each airline company (Only Top 10 Row )
> flights %>%
+   filter (year == 2013) %>%
+   mutate (speed_mph = distance / (hour + (minute/60))) %>%
+   inner_join(airlines,by = "carrier") %>%
+   group_by(name,origin,dest) %>%
+   summarise(mean_speed = mean(speed_mph)) %>%
+   arrange(name,desc(mean_speed)) %>%
+   head(10)
`summarise()` has grouped output by 'name', 'origin'. You can override using the `.`groups` argument.
# A tibble: 10 × 4
# Groups:   name, origin [3]
  name      origin dest  mean_speed
  <chr>     <chr> <chr>    <dbl>
1 AirTran Airways Corporation LGA  MKE    101.
2 AirTran Airways Corporation LGA  ATL    65.4
3 AirTran Airways Corporation LGA  CAK    28.1
4 Alaska Airlines Inc.       EWR  SEA    227.
5 American Airlines Inc.    JFK  SFO    224.
6 American Airlines Inc.    JFK  LAS    211.
7 American Airlines Inc.    JFK  STT    203.
8 American Airlines Inc.    JFK  LAX    195.
9 American Airlines Inc.    JFK  SJU    159.
10 American Airlines Inc.   JFK  SAN   140.

```



Q5: Which airline has the least delay time, ranked from the least to the most?

```

flights %>%
  filter (year == 2013) %>%
  inner_join(airlines,by = "carrier") %>%
  group_by(name) %>%
  summarise(Total_Delay = sum(dep_delay,na.rm = TRUE)) %>%
  arrange(Total_Delay) %>%
  head(10)

```

Result Set

```

> ## Q5 : Which airports have the minimum delay time, arranged in descending order?
> flights %>%
+   filter (year == 2013) %>%
+   inner_join(airlines,by = "carrier") %>%
+   group_by(name) %>%
+   summarise(Total_Delay = sum(dep_delay,na.rm = TRUE)) %>%
+   arrange(Total_Delay) %>%
+   head(10)
# A tibble: 10 × 2
  name          Total_Delay
  <chr>        <dbl>
1 SkyWest Airlines Inc.      365
2 Hawaiian Airlines Inc.    1676
3 Alaska Airlines Inc.     4133
4 Mesa Airlines Inc.       10353
5 Frontier Airlines Inc.   13787
6 AirTran Airways Corporation 59680
7 Virgin America            66033
8 US Airways Inc.          75168
9 Southwest Airlines Co.   214011
10 Envoy Air                265521
> |

```

▼ Assign Homework (HW2)

- HW 2 - Restaurant pizza SQL
 - create 3 - 5 data frames ⇒ write table into server.



Preparing Pizza Project convert to PostgresDB

- Script from SQLListDB()

pizza.db

- Import DB to postit



Script Transform to PostgreSQL

Assignment2_MigrateToPost.R

```

## Connect db
library(RPostgreSQL)
library(RSQLite)
library(glue)
library(tidyverse)

## create connection to sqlite.db file
con <- dbConnect(SQLite(),"pizza.db")
con_pros <- dbConnect(
  PostgreSQL(),
  host = "floppy.db.elephantsql.com",
  dbname = "tuhylmk1",
  user = "tuhylmk1",
  password = "QHYXeMm5-MPB6hEvb9F7FBlus_Q0bPKL",
  port = 5432)
##list Table
dbListTables(con)
dbListTables(con_pros)

##list fields/columns

dbListFields(con, "CUSTOMER")
dbListFields(con, "country")
dbListFields(con, "Ingredient")
dbListFields(con, "Menu_Pizza")
dbListFields(con, "ORDERS")
dbListFields(con, "Order_Pizza")
dbListFields(con, "Pizza_Ingredient")

## get data from database tables

customer <- data.frame(dbGetQuery(con, "select * from customer"))
country <- data.frame(dbGetQuery(con, "select * from country"))
ingredient <- data.frame(dbGetQuery(con, "select * from Ingredient"))

```

```

menu_Pizza <- data.frame(dbGetQuery(con, "select * from Menu_
orders <- data.frame(dbGetQuery(con, "select * from ORDERS"))
order_pizza <- data.frame(dbGetQuery(con, "select * from Order_
pizza_ingredient <- data.frame(dbGetQuery(con, "select * from Pizza_Ingredient

## write table to database
dbWriteTable(con_pros,"customer",customer)
dbWriteTable(con_pros,"country",country)
dbWriteTable(con_pros,"ingredient",ingredient)
dbWriteTable(con_pros,"menu_pizza",menu_Pizza)
dbWriteTable(con_pros,"orders",orders)
dbWriteTable(con_pros,"order_pizza",order_pizza)
dbWriteTable(con_pros,"pizza_ingredient",pizza_ingredient)

dbListTables(con_pros)

dbGetQuery(con_pros, "select * from customer")
dbGetQuery(con_pros, "select * from country")
dbGetQuery(con_pros, "select * from ingredient")
dbGetQuery(con_pros, "select * from menu_pizza")
dbGetQuery(con_pros, "select * from orders")
dbGetQuery(con_pros, "select * from order_pizza")
dbGetQuery(con_pros, "select * from pizza_ingredient")

dbDisconnect(con)
dbDisconnect(con_pros)

```

Result Set

```

> ## Connect db
> library(RPostgreSQL)
> library(RSQLite)
> library(glue)
> library(tidyverse)
>
> ## create connection to sqlite.db file
> con <- dbConnect(SQLite(),"pizza.db")
> con_pros <- dbConnect(
+   PostgreSQL(),
+   host = "floppy.db.elephantsql.com",
+   dbname = "tuhylmkl",
+   user = "tuhylmkl",
+   password = "QHYXeMm5-MPB6hEvb9F7FBlus_Q0bPKL",
+   port = 5432)
> ##list Table
> dbListTables(con)
[1] "CUSTOMER"          "Country"           "Ingredient"        "Menu_Pizza"       "ORDERS"
[6] "Order_Pizza"        "Pizza_Ingredient"
> dbListTables(con_pros)
character(0)
>

```

```

> ##list fields/columns
>
> dbListFields(con,"CUSTOMER")
[1] "CustomerID"          "CustNm"            "email"             "BirthDate"         "Customer_CountryID"
> dbListFields(con,"country")
[1] "CountryID" "CountryNm"
> dbListFields(con,"Ingredient")
[1] "IngredID"    "IngredName"      "No_Stock_Remain" "Min_Stock"
> dbListFields(con,"Menu_Pizza")
[1] "PizzaID"     "Pizza_Desc"     "Price_Per_Each"
> dbListFields(con,"ORDERS")
[1] "OrderID"     "OrderDate"      "CustomerID"      "Total_Amount"    "Delivery_Type"
> dbListFields(con,"Order_Pizza")
[1] "OrderID"     "PizzaID"
> dbListFields(con,"Pizza_Ingredient")
[1] "PizzaID"     "IngredID"
>

```

```

> ## get data from database tables
>
> customer <- data.frame(dbGetQuery(con, "select * from customer"))
> country <- data.frame(dbGetQuery(con, "select * from country"))
> ingredient <- data.frame(dbGetQuery(con, "select * from Ingredient"))
> menu_Pizza <- data.frame(dbGetQuery(con, "select * from Menu_Pizza"))
> orders <- data.frame(dbGetQuery(con, "select * from ORDERS"))
> order_pizza <- data.frame(dbGetQuery(con, "select * from Order_Pizza"))
> pizza_ingredient <- data.frame(dbGetQuery(con, "select * from Pizza_Ingredient"))
>
> ## write table to database
> dbWriteTable(con_pros,"customer",customer)
[1] TRUE
> dbWriteTable(con_pros,"country",country)
[1] TRUE
> dbWriteTable(con_pros,"ingredient",ingredient)
[1] TRUE
> dbWriteTable(con_pros,"menu_pizza",menu_Pizza)
[1] TRUE
> dbWriteTable(con_pros,"orders",orders)
[1] TRUE
> dbWriteTable(con_pros,"order_pizza",orders)
[1] TRUE
> dbWriteTable(con_pros,"pizza_ingredient",orders)
[1] TRUE
>
> dbListTables(con_pros)
[1] "customer"          "country"           "ingredient"        "order_pizza"      "menu_pizza"
[6] "orders"            "pizza_ingredient"
>

```

```

> dbGetQuery(con_pros, "select * from customer")
  row.names CustomerID CustNm           email BirthDate Customer_CountryID
1             1   Jenny jenny@gmail.com 01/01/2000                  1
2             2   Henry Henry@gmail.com 10/05/1996                  2
3             3  Marry Marry@gmail.com 27/12/1900                  3
4             4   John John@gmail.com 10/04/1997                  1
5             5  Jeffy Jeffy@gmail.com 10/05/2002                  3
6             6  Gammy Gammy@gmail.com 20/06/2019                  2
7             7   Anna Anna@gmail.com 01/02/1997                  4
8             8  Pakpum Pakpum@gmail.com 20/08/1995                  1
9             9  Creamy Creamy@gmail.com 10/05/1994                  2
> dbGetQuery(con_pros, "select * from country")
  row.names CountryID CountryNm
1             1    Thailand
2             2     Canada
3             3      Japan
4             4       USA
> dbGetQuery(con_pros, "select * from ingredient")
  row.names IngredID IngredName No_Stock_Remain Min_Stock
1             1     Ing001    Onion          20            5
2             2     Ing002     Ham          10            3
3             3     Ing003  Mushroom         25            2
4             4     Ing004    Bacon          40           10
5             5     Ing005   Chilli          10            2
6             6     Ing006  Sausage          60            5
7             7     Ing007 Pineapple         40            3

```

```

> dbGetQuery(con_pros, "select * from menu_pizza")
  row.names PizzaID Pizza_Desc Price_Per_Each
1             1   Pizza1    Pizza Art        250
2             2   Pizza2   Cheesy Pizza       650
3             3   Pizza3   Yummy Pizza       890
4             4   Pizza4 Healthy Pizza       255
5             5   Pizza5   Pizza Oven        235
6             6   Pizza6   Bacon Pizza       650
7             7   Pizza7 Cooking Pizza       235
8             8   Pizza8 Homemade Pizza      125
9             9   Pizza9   Good Pizza        250

```

```

> dbGetQuery(con_pros, "select * from orders")
  row.names OrderID OrderDate CustomerID Total_Amount Delivery_Type
1             1 0001 11/05/2002        4       890 At Store
2             2 0002 20/06/2002        5       700 Delivery
3             3 0003 30/04/2002        6       650 Delivery
4             4 0004 05/06/2002        6      1020 Delivery
5             5 0005 15/03/2001        3       650 At Store
6             6 0006 16/07/2002        2       780 Delivery
7             7 0007 10/04/2001        9       900 Delivery
8             8 0008 06/08/2002        6       660 At Store
9             9 0009 19/03/2002        8       800 Delivery
10            10 0010 25/04/2001       7      1200 At Store
11            11 0011 30/05/2002       2       450 At Store
12            12 0012 07/08/2001       1       600 At Store
13            13 0013 05/09/2002       3       750 At Store
14            14 0014 08/05/2002       8       900 Delivery
15            15 0015 06/04/2002       4      1200 Delivery

```

```
> dbGetQuery(con_pros, "select * from order_pizza")
  row.names OrderID OrderDate CustomerID Total_Amount Delivery_Type
1           1 11/05/2002          4        890    At Store
2           2 20/06/2002          5        700    Delivery
3           3 30/04/2002          6        650    Delivery
4           4 05/06/2002          6       1020    Delivery
5           5 15/03/2001          3        650    At Store
6           6 16/07/2002          2        780    Delivery
7           7 10/04/2001          9        900    Delivery
8           8 06/08/2002          6        660    At Store
9           9 10/03/2002          8        800    Delivery
10         10 25/04/2001          7       1200    At Store
11         11 30/05/2002          2        450    At Store
12         12 07/08/2001          1        600    At Store
13         13 05/09/2002          3        750    At Store
14         14 08/05/2002          8        900    Delivery
15         15 06/04/2002          4       1200    Delivery
```

```
> dbGetQuery(con_pros, "select * from pizza_ingredient")
  row.names OrderID OrderDate CustomerID Total_Amount Delivery_Type
1           1 11/05/2002          4        890    At Store
2           2 20/06/2002          5        700    Delivery
3           3 30/04/2002          6        650    Delivery
4           4 05/06/2002          6       1020    Delivery
5           5 15/03/2001          3        650    At Store
6           6 16/07/2002          2        780    Delivery
7           7 10/04/2001          9        900    Delivery
8           8 06/08/2002          6        660    At Store
9           9 10/03/2002          8        800    Delivery
10         10 25/04/2001          7       1200    At Store
11         11 30/05/2002          2        450    At Store
12         12 07/08/2001          1        600    At Store
13         13 05/09/2002          3        750    At Store
14         14 08/05/2002          8        900    Delivery
15         15 06/04/2002          4       1200    Delivery
> dbListTables(con_pros)
[1] "customer"          "country"          "ingredient"        "order_pizza"      "menu_pizza"
[6] "orders"             "pizza_ingredient"
```