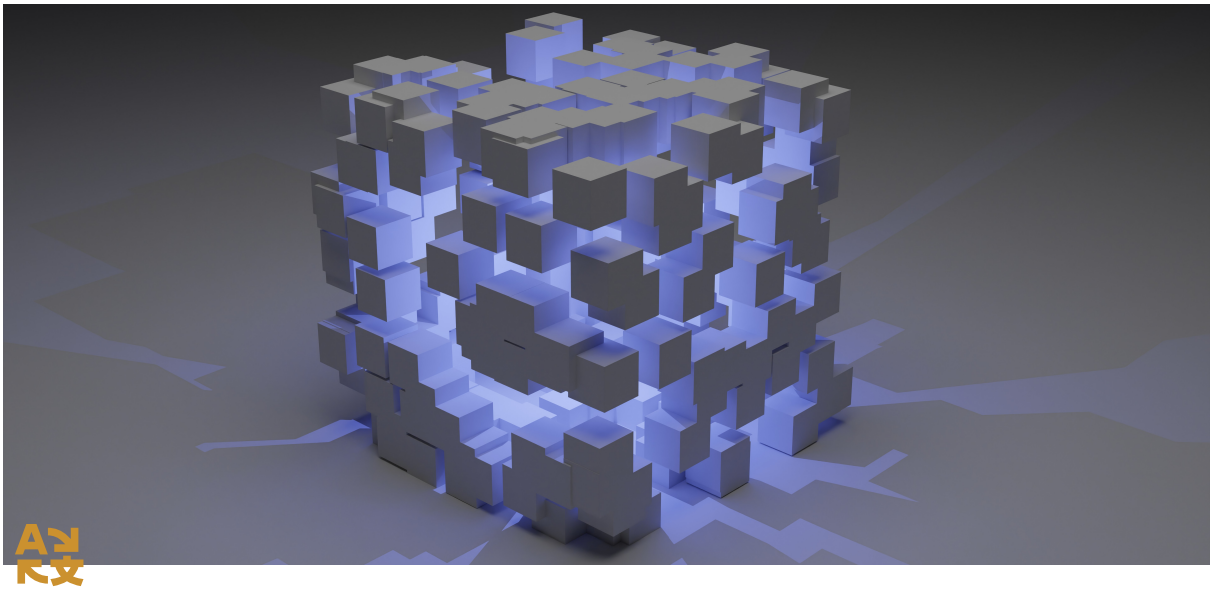
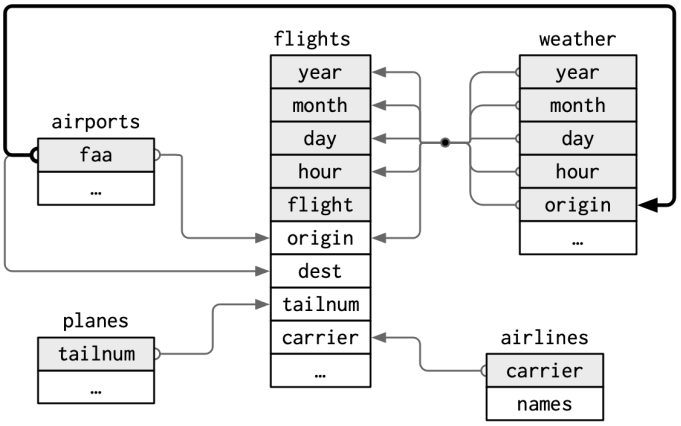


04 - Homework: Data Transformation

Tags	Homework
Lesson	04 - Data Transformation
Status	Done

HOMEWORK 1: nycflights13



ER Diagram for nycflights13

Dataset preparation ⇒ packages ⇒ install `nycflights13`

Data transformation (packages nycflight13)
by ⇒ ask 5+ questions about this dataset

Functions: select, filter, arrange, mutate, summarize, count, etc.

Prepare packages and data

```
# Check if not install packages yet
if (!require(tidyverse)) {
  install.packages("tidyverse")
}

if (!require(dplyr)) {
  install.packages("dplyr")
}

if (!require(nycflights13)) {
  install.packages("nycflights13")
}

# call for packages
library(tidyverse)
library(dplyr)
library(nycflights13)

# data set
data("airlines")
data("airports")
data("flights")
data("planes")
data("weather")
```

Q1: [Top15] Average delay time for each airline?

```r
## Q1. Average delay time for each airline?
# join dataset [airline <-> flights]
q1_joined_data <- flights %>%
  left_join(airlines, by = "carrier")

#cal avg arrival delay and group data
avg_delay_each_airline <- q1_joined_data %>%
  group_by(name) %>%
  mutate(arr_delay = replace_na(arr_delay, 0)) %>%
  summarize(avg_delay = mean(arr_delay)) %>%
  arrange(desc(avg_delay)) %>%
  head(15)

print(avg_delay_each_airline)
```

```
> print(avg_delay_each_airline)
# A tibble: 15 × 2
   name                          avg_delay
   <chr>                             <dbl>
 1 Frontier Airlines Inc.            21.8
 2 AirTran Airways Corporation       19.6
 3 ExpressJet Airlines Inc.          14.9
 4 Mesa Airlines Inc.                14.1
 5 SkyWest Airlines Inc.             10.8
 6 Envoy Air                         10.2
 7 Southwest Airlines Co.             9.47
 8 JetBlue Airways                    9.36
 9 Endeavor Air Inc.                  6.91
10 United Air Lines Inc.              3.50
11 US Airways Inc.                    2.06
12 Virgin America                     1.75
13 Delta Air Lines Inc.               1.63
14 American Airlines Inc.             0.356
15 Hawaiian Airlines Inc.            -6.92
```

🤑 **BENEFITS:**

- identify `areas for improvement` in operational efficiency and customer satisfaction.
- `prioritize efforts` to reduce delay times for the airlines that consistently have the highest delays.

Q2: [Top10] Frequency & number of flights vary across different seasons?

```r
## Q2. Frequency & number of flights vary across diff
# Filter the data to include only rows with non-NA va

data("flights")

flights <- flights %>%
  filter(!is.na(origin) & !is.na(dest) & !is.na(month

# Define the 'season' variable based on 'month'
flights_grouped <- flights %>%
  mutate(season = case_when(
    month %in% c(12, 1, 2) ~ "Winter",
    month %in% c(3, 4, 5) ~ "Spring",
    month %in% c(6, 7, 8) ~ "Summer",
    month %in% c(9, 10, 11) ~ "Fall"
  ))

# Group flights by origin, destination, and season
flights_grouped <- flights_grouped %>%
  group_by(origin, dest, season) %>%
  summarize(Total_flights = n())

# Convert the 'season' column into separate columns
flights_grouped <- flights_grouped %>%
  spread(season, Total_flights, fill = 0)

# Sum for each origin and dest
flights_grouped <- flights_grouped %>%
```

```
> # Print the updated data frame
> print(flights_grouped)
# A tibble: 10 × 7
# Groups:   origin [3]
   origin dest  Total_flights  Fall Spring Summer Winter
   <chr>  <chr>         <dbl> <dbl>  <dbl>  <dbl>  <dbl>
 1 JFK    LAX           11262  2797   2855   2892   2718
 2 LGA    ATL           10263  2486   2706   2533   2538
 3 LGA    ORD            8857  2474   2225   2384   1774
 4 JFK    SFO            8204  2105   2025   2108   1966
 5 LGA    CLT            6168  1712   1412   1672   1372
 6 EWR    ORD            6100  1515   1533   1697   1355
 7 JFK    BOS            5898  1454   1466   1550   1428
 8 LGA    MIA            5781  1447   1441   1443   1450
 9 JFK    MCO            5464  1278   1422   1437   1327
10 EWR    BOS            5327  1364   1345   1402   1216
```

```
  group_by(origin, dest) %>%
  summarize(
    Fall = sum(Fall),
    Spring = sum(Spring),
    Summer = sum(Summer),
    Winter = sum(Winter),
    Total_flights = sum(Fall, Spring, Summer, Winter)
    )
# Reorder the columns to match the desired order
flights_grouped <- flights_grouped %>%
  select(origin, dest, Total_flights, Fall, Spring, S
  arrange(desc(Total_flights)) %>%
  head(10)

# Print the updated data frame
print(flights_grouped)
```

🤑 **BENEFITS:**

- Understanding `seasonal trends` in flight demand can help airlines allocate their resources more efficiently.
- Uncovering patterns in route popularity can inform `marketing campaigns and promotional efforts`.
- Decision making about expanding their `network` or adding new `routes`.

Q3: What are the top 15 routes by total number of flights, considering both wide-body and narrow-body aircraft?

```
# Q3. What are the top 15 routes by total number of f
# considering both wide-body and narrow-body aircraft

data("flights")
data("planes")

# Cal avg number of seats
avg_seats <- mean(planes$seats)
# Categorize into wide or narrow->planes
#assume cutoff point is average of seats
planes$aircrafts_type <- ifelse(planes$seats > avg_se

# Calculate the total number of flights for each rout
route_counts <- flights %>%
  left_join(planes, by = "tailnum") %>%
  group_by(origin, dest, aircrafts_type) %>%
  count() %>%
  rename(total_flights = n)

route_counts <- route_counts %>%
  filter(!is.na(aircrafts_type))

# Calculate the average distance flown for each route
avg_distance_per_route <- flights %>%
  left_join(planes, by = "tailnum") %>%
```

	origin	dest	avg_distance	total_flights	aircrafts_type
	<chr>	<chr>	<dbl>	<int>	<chr>
1	JFK	LAX	2475	11065	Wide_body
2	JFK	SFO	2586	7574	Wide_body
3	EWR	ATL	746	4608	Narrow_body
4	LGA	ATL	762	4496	Narrow_body
5	LGA	ORD	733	4477	Wide_body
6	JFK	MCO	944	4016	Wide_body
7	LGA	DCA	214	3937	Wide_body
8	EWR	MCO	937	3805	Wide_body
9	LGA	BOS	184	3773	Narrow_body
10	JFK	SJU	1598	3511	Wide_body
11	LGA	CLT	544	3472	Wide_body
12	JFK	BOS	187	3432	Narrow_body
13	EWR	BOS	200	3409	Narrow_body
14	JFK	LAS	2248	3367	Wide_body
15	JFK	FLL	1069	3361	Wide_body

```
  group_by(origin, dest, aircrafts_type) %>%
  summarize(avg_distance = mean(distance))

# Join the route counts and average distance data
route_summary <- route_counts %>%
  left_join(avg_distance_per_route, by = c("origin",
  arrange(desc(total_flights))

# Identify the top 15 routes
top_15_routes <- route_summary %>%
  select(origin, dest, avg_distance, total_flights, a
  head(15)

# Print the top 15 routes with average distance flown
print(top_15_routes)
# if we use this pattern, we still can report full li
```

🤑 **BENEFITS:**

- `Optimizing` flight schedules and resource allocation.
- Average distance flown for each route → assess the `fuel efficiency` of different aircraft types.

Q4: How does the average arrival delay differ between flights with and without visibility data?

```
### Assuming no weather data is available for compari
### Instead of relying on missing values, a cutoff pc
# Q4. How does the average arrival delay differ betwe

data("flights")
data("weather")
data("planes")

# Filter for delayed flights
delayed_flights <- flights %>%
  filter(arr_delay > 0)

# Join data frames based on origin, year, month, and
delayed_flights_with_weather <- delayed_flights %>%
  left_join(weather, by = c("origin", "year", "month"
  select(dep_time, arr_time, arr_delay, dep_delay, vi

# Identify flights with no weather data
delayed_flights_no_weather <- delayed_flights %>%
  left_join(weather, by = c("origin", "year", "month"
  filter(is.na(visib))

# Calculate average arrival delay for both groups by
avg_delay_with_weather_by_model <- delayed_flights_wi
  left_join(planes, by = "tailnum") %>%
```

```
> print(summary_table)
       model delay_difference avg_visib
1   A330-223        218.00000 10.000000
2       G-IV        146.50000  9.750000
3    757-351        143.00000  9.937500
4    747-451         96.75000  1.399583
5    737-990         67.29474  9.096053
6  767-432ER         66.67568  8.813063
7   A319-115         65.71140  9.173691
8        150         60.44885  9.150313
9      A109E         58.38206  9.170343
10      A-1B         55.12500  9.262500
>
```

> This code should rewrite for more correct and more accurate result

```
  group_by(model) %>%
  summarize(avg_delay_with_weather = mean(arr_delay))

avg_delay_no_weather_by_model <- delayed_flights_no_w
  left_join(planes, by = "tailnum") %>%
  group_by(model) %>%
  summarize(avg_delay_no_weather = mean(arr_delay))

# Calculate delay difference by model
delay_difference_by_model <- avg_delay_with_weather_b
  avg_delay_no_weather_by_model$avg_delay_no_weather

# Calculate average visibility by model
avg_visib_by_model <- delayed_flights_with_weather %>
  filter(!is.na(visib)) %>%
  left_join(planes, by = "tailnum") %>%
  group_by(model) %>%
  summarize(avg_visib = mean(visib))

# Combine model and delay difference data into a data
summary_table <- data.frame(model = avg_delay_with_we
                            delay_difference = delay_
                            avg_visib = avg_visib_by_
  arrange(desc(delay_difference_by_model)) %>%
  head(10)
# Print the summary table
print(summary_table)
```

🤑 **BENEFITS:**

- Compare the `impact of weather` on flight delays for different types of aircraft.
- Identify the aircraft models that are most `affected by weather` delays.

Q5: What is the most popular origin airport for flights departing during the summer months?

```
# Q5. What is the most popular origin airport for fli

# Filter flights departing during (June, July, August
summer_flights <- flights %>%
  filter(month %in% c(6, 7, 8))

# Count the number of flights for each origin airport
origin_count_summer <- summer_flights %>%
  group_by(origin) %>%
  summarise(flight_count = n())

# Sort the results in descending order of flight coun
sorted_origin_count_summer <- origin_count_summer %>%
  left_join(airports, by = c("origin" = "faa")) %>%
```

```
> print(head(sorted_origin_count_summer))
# A tibble: 3 × 5
  name                flight_count tzone             tz dst
  <chr>                      <int> <chr>          <dbl> <chr>
1 Newark Liberty Intl        31009 America/New_York    -5 A
2 John F Kennedy Intl        29478 America/New_York    -5 A
3 La Guardia                 26508 America/New_York    -5 A
>
```

```
  select(name, flight_count, tzone, tz, dst) %>%
  arrange(desc(flight_count))

# Print the most popular origin airport
print(head(sorted_origin_count_summer))
```

🤑 **BENEFITS:**

- `Resource allocation`, `capacity planning`, and marketing campaigns.
- Identify trends in `travel patterns`.

HOMEWORK 2: Restaurant pizza SQL

HOMEWORK2.R

CSV_pack.zip

upload database to PostgresSQL server on ElpphantSQL

https://www.elephantsql.com/

⇒ Create 3-5 dataframes

⇒ Write table into server

Prepare library

```
## connect to PostgreSQL server
library(RPostgreSQL)
library(tidyverse)
```

Create connection to `elephantsql.com`

```
## create connection
con <- dbConnect(
  PostgreSQL(),
  host =    "arjuna.db.elephantsql.com",
  dbname = "equcxncr",
  user = "equcxncr",
  password = "XXXXXXXXXXXXXXXXXXXXXXXXXXXXX",
  port = 5432 #default port for PostgreSQL
)
```

Create database table

```
## Data from Pizza Restaurant in SQL Homework

## create 5 table from SQL homework
# dataframe "customers"
df_customers <- data.frame(
  customer_id = c(1:10),
  first_name = c("Somchai", "Jitrlada", "Pichai", "Sirin", "Wichai", "Kanokwan", "Chaiwa
  last_name = c("Churnsri", "Sitthiphant", "Phirom", "Maneerat", "Wongwichit", "Pongsati
  phone_number = c("089-123-4567", "098-765-4321", "086-543-2109", "097-654-3210", "088-
  email = c("somchai@gmail.com", "jitlada@hotmail.com", "pichai@yahoo.com", "sirinth@out
)

# dataframe "orders"
df_orders <- data.frame(
  order_id = c(1:10),
  customer_id = c(1, 1, 2, 3, 4, 4, 6, 9, 9, 10),
  menu_id = c(4, 4, 3, 4, 5, 6, 7, 4, 9, 10),
  branch_id = c(2, 2, 3, 4, 5, 6, 2, 8, 2, 10),
  order_date = strptime(
    c("2023-11-05 10:28:01",
      "2023-11-06 12:28:01",
      "2023-11-07 14:28:01",
      "2023-11-08 15:28:01",
      "2023-11-09 19:28:01",
      "2023-11-10 15:28:01",
      "2023-11-11 09:38:01",
      "2023-11-12 15:28:01",
      "2023-11-13 15:28:01",
      "2023-11-14 15:28:01"),
    "%Y-%m-%d %H:%M:%S"),
  quantity = c(1, 2, 3, 4, 5, 6, 7, 8, 9, 10),
  total_amount = c(299.50, 598.75, 897.00, 1196.00, 1495.00, 1794.25, 2093.00, 2392.00,
)

# dataframe "menus"
df_menus <- data.frame(
  menu_id = c(1:10),
  menu_name = c("Pizza with Pepperoni", "Hawaiian Pizza", "Pizza with Sausage", "Pizza w
  meu_size = c("14 inches", "10 inches", "16 inches", "12 inches", "12 inches", "16 inch
  menu_price = c(399.00, 249.00, 499.00, 399.00, 299.00, 499.00, 399.00, 249.00, 349.00,
  menu_category = c("Classic Pizza", "Fruit Pizza", "Meat Pizza", "Seafood Pizza", "Vege
)

# dataframe "branches"
df_branches <- data.frame(
  branch_id = c(1:11),
  branch_name = c("Ratchadaphisek Branch", "Rama IX Branch", "Pattaya Branch", "Phuket B
  branch_address = c(
    "1213 Ratchadaphisek Road, Bangkok 10310",
    "1415 Rama IX Road, Bangkok 10240",
    "1617 Pattaya Nuea Road, Pattaya, Chonburi 20150",
    "1819 Ratsada Road, Phuket Town, Phuket 83000",
    "2021 Charoenmuang Road, Chiang Mai City, Chiang Mai 50200",
    "2223 Mittraphap Road, Nakhon Ratchasima City, Nakhon Ratchasima 30000",
```

```
    "1111 Naresuan Road, Ayutthaya City, Ayutthaya 13000",
    "2222 Nakhon Kaen Road, Khon Kaen City, Khon Kaen 40000",
    "3333 Udon Thani Road, Udon Thani City, Udon Thani 41000",
    "4444 Chiang Rai Road, Chiang Rai City, Chiang Rai 57000",
    "5555 Universe Road, Metro City, Metrospace 90000"),
  bracnh_city = c("Bangkok", "Bangkok", "Pattaya", "Phuket", "Chiang Mai", "Nakhon Ratch
  branch_country = c("Thailand", "Thailand", "Thailand", "Thailand", "Thailand", "Thaila
)

# dataframe "pizza_ingredient"
df_pizza_ingredient <- data.frame(
  ingredient_id = c(1:10),
  menu_id = c(1, 1, 1, 2, 2, 2, 3, 3, 3, 4),
  ingredient_name = c("Tomato Sauce", "Mozzarella Cheese", "Pepperoni", "Tomato Sauce",
  ingredient_category = c("Seasoning", "Topping", "Meat", "Seasoning", "Topping", "Meat"
  ingredient_price =c(10.00, 20.00, 30.00, 10.00, 20.00, 30.00, 10.00, 20.00, 30.00, 10.
)
```

Write table into Cloud database

```
# convert to vector for code refactor in dbWriteTable
table_names <- c("customers", "orders", "menus", "branches", "pizza_ingredient")
dataframes <- list(df_customers, df_orders, df_menus, df_branches, df_pizza_ingredient)

# Write data to the tables
for (i in 1:length(table_names)) {
  dbWriteTable(con, table_names[i], dataframes[[i]], row.names = F)
}

## doublecheck table list by using dbListTables
dbListTables(con)
```

Get query from Cloud database (3 questions)

1. top 4 customers w/ the highest spending

```
# 1. top 4 customers w/ the highest spending

Q1 <- dbGetQuery(con, "
WITH customer_spending AS (
  SELECT
    customers.customer_id,
    SUM(orders.total_amount) AS total_spending
  FROM customers
  JOIN orders ON customers.customer_id = orders
  GROUP BY customers.customer_id
)

SELECT
  customers.first_name,
  customers.last_name,
  customer_spending.total_spending,
  email
```

```
> cat("Top 4 customers with the highest spending\n")
Top 4 customers with the highest spending
> Q1
  first_name   last_name total_spending                email
1    Worawit Chanprasert        5083.00    worawit@icloud.com
2      Sirin    Maneerat        3289.25   sirinth@outlook.com
3    Pimolan    Sriwarah        2990.00    pimolwan@gmail.com
4   Kanokwan   Pongsatit        2093.00 kanokwan@tutanota.com
```

```
FROM customers
JOIN customer_spending ON customers.customer_id
ORDER BY customer_spending.total_spending DESC
LIMIT 4;")

cat("Top 4 customers with the highest spending\
Q1
```

2. Which branch has the highest order count / highest total revenues? (between 5 November and 14 November)

```
# 2. Which branch has the highest order count /
# (between 5 November and 14 November)

Q2 <- dbGetQuery(con, "
WITH latest_orders AS (
  SELECT
    branch_id,
    MAX(order_date) AS latest_order_date
  FROM orders
  GROUP BY branch_id
)

SELECT
  branches.branch_name,
  COUNT(*) AS order_count,
  SUM(orders.total_amount) AS total_revenues,
  latest_orders.latest_order_date
FROM orders
JOIN branches ON orders.branch_id = branches.br
JOIN latest_orders ON orders.branch_id = latest_
WHERE order_date BETWEEN '2023-11-05 18:00:00' A
GROUP BY branch_name, latest_orders.latest_orde
ORDER BY order_count DESC;
")

cat("Which branch has the highest order count /
Q2
```

```
> cat("Which branch has the highest order count / highest total revenues? (betwee
n 5 november and 14 november)\n")
Which branch has the highest order count / highest total revenues? (between 5 nov
ember and 14 november)
> Q2
             branch_name order_count total_revenues   latest_order_date
1          Rama IX Branch           3        5382.75 2023-11-13 15:28:01
2       Chiang Rai Branch           1        2990.00 2023-11-14 15:28:01
3        Khon Kaen Branch           1        2392.00 2023-11-12 15:28:01
4       Chiang Mai Branch           1        1495.00 2023-11-09 19:28:01
5          Pattaya Branch           1         897.00 2023-11-07 14:28:01
6           Phuket Branch           1        1196.00 2023-11-08 15:28:01
7 Nakhon Ratchasima Branch          1        1794.25 2023-11-10 15:28:01
>
```

3. Which menu is most popular -> each branch?

```
Q3 <- dbGetQuery(con,"
SELECT
  menus.menu_name,
  branches.branch_name,
  COUNT(*) AS order_count
FROM orders
JOIN branches ON orders.branch_id = branches.br
JOIN menus ON orders.menu_id = menus.menu_id
GROUP BY branch_name, menu_name
ORDER BY order_count DESC;")

cat("Which menu is most popular -> each branch?
Q3
```

```
> cat("Which menu is most popular -> each branch?\n")
Which menu is most popular -> each branch?
> Q3
                   menu_name             branch_name order_count
1          Pizza with Seafood          Rama IX Branch           2
2        Pizza with Pineapple          Rama IX Branch           1
3          Pizza with Seafood           Phuket Branch           1
4          Pizza with Seafood         Khon Kaen Branch           1
5          Pizza with Sausage           Pattaya Branch           1
6      Pizza with BBQ Chicken Nakhon Ratchasima Branch           1
7          Pizza with Bacon        Chiang Rai Branch           1
8 Pizza with Mixed Vegetables        Chiang Mai Branch           1
9     Pizza with Grilled Pork          Rama IX Branch           1
>
```

⛔Disconnect from `elephantsql.com`

```
## close connection
dbDisconnect(con)
```

Write results to CSV flies

```
## Export results to CSV files
write.csv(Q1, "Q1.csv", row.names = F)
write.csv(Q2, "Q2.csv", row.names = F)
write.csv(Q3, "Q3.csv", row.names = F)
```