COMP 120: Revision Questions for Test 2 (Part B)

# Overview

The goal of the practical test is to evaluate your ability to:

1. perform data manipulation operations
2. perform data aggregation operations
3. perform data reshaping operations
4. perform web scraping operations
5. handling relational data using relational operations
6. perform data modelling – linear models and Random Forest (only)
7. perform reading and writing of files using read\_csv() and write\_csv() functions – note these are helper functions you will use in the context of the operations indicated above.
8. create visualisations using ggplot()
9. perform operations using other useful functions: is.na, ifelse and cut

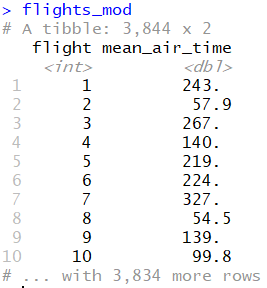
The best way to prepare for this test is to go through all the lecture slides (lectures 8, 10-20), lab tasks (labs 4-9) and the associated mastery tasks (mastery 4-9). Note that demo code is provided for each lecture and you will benefit from understanding and executing the code.

The following sample questions will help you understand the nature of practical test questions – however, ***this is not an exhaustive list***. Attempt these questions after you have had a chance to revise all the course material (lectures, labs and mastery tasks).

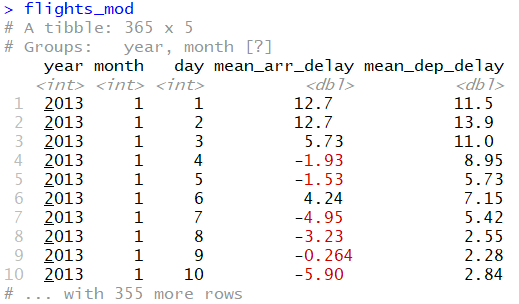
# Questions

Use pipes to combine commands wherever possible while answering the questions below.

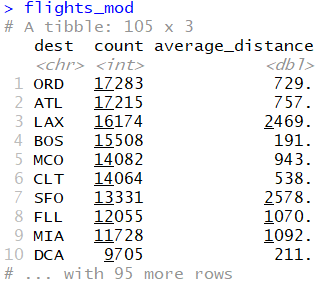
1. Using the *flights* dataset answer the following questions. You must load *tidyverse* and *nycflights13* packages before you can answer these questions. The flights dataset comes in the nycflights13 package.
   1. Create a new column called air\_time\_seconds which converts air\_time in minutes to seconds. Then select the following columns and store it in a new variable called flights\_mod: year, month, day, flight, air\_time and air\_time\_seconds.
   2. Using the result from the previous question, display the average air\_time per flight as shown below. Store the result in flights\_mod.



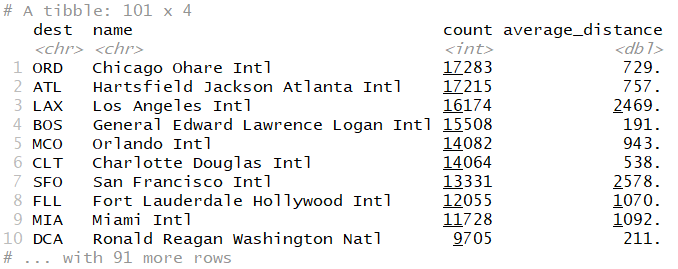
* 1. Using data from the *flights* dataset, display the average departure delay and arrival delay for all flights grouped by year, day and month (see output below). Store the result in flights\_mod.



* 1. Using data from the *flights* dataset, display the counts, and average distance travelled for all destinations arranged by descending order of counts (see output below). Store the result in a variable called flights\_mod.



* 1. The details about airports can be found in the *airports* dataset that comes with nycflights13 package. You can view the details of the airports by typing *airports* command. Use *?airports* to see the details of all the columns. Now, use the output from the previous step (flights\_mod) and the *airports* dataset to produce the following result. Hint: The values in *dest* column in flights tibble and *faa* column in airports tibble should be matched. See the correct syntax in slide 29 of lecture 15.



1. Examine table4a that comes built-in with tidyverse. Convert the data into the output shown below in a tibble called *table4a\_mod*.



Now, write code to convert the data in table4a\_mod so that the data looks like table4a. Store this result in a tibble called *table4a\_original*.

1. Assume you have been given the following code about stocks: year, quarter (qtr variable) and the rate of return (return variable).

stocks <- **tibble**(

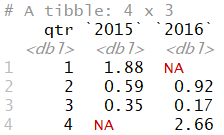
year = **c**(2015, 2015, 2015, 2015, 2016, 2016, 2016),

qtr = **c**( 1, 2, 3, 4, 2, 3, 4),

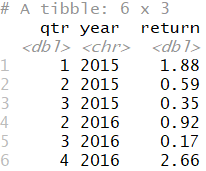
return = **c**(1.88, 0.59, 0.35, NA, 0.92, 0.17, 2.66)

)

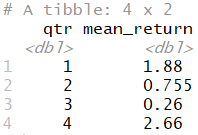
a. Convert the data in the stocks tibble into the following and store it in a tibble called stocks\_mod:



b. Now, reshape stocks\_mod into the following:



c. Finally, aggregate the results by computing the mean of the results above based on quarter variable.



1. Data for three tibbles are given below: doctor, diagnosis and patient. Answer the questions given below using join functions. The doctor table has doctor’s code and name. The patient table contains patient’s id and name. The diagnosis table contains patient’s id, doctor’s code and the diagnosed illness.

doctor <- tibble(

dcode = c("d1", "d2"),

dname = c("Alice", "Bob")

)

diagnosis <- tibble(

pid = c("p1", "p2", "p3", "p4"),

dcode = c("d4", "d1", "d2", "d2"),

illness = c("Jaundice", "Peptic ulcer", "Conjunctivitis", "Tonsillitis")

)

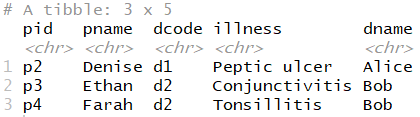
patient <- tibble(

pid = c("p1", "p2", "p3", "p4"),

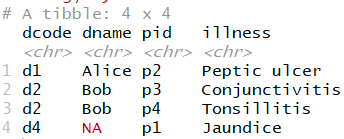
pname = c("Carol", "Denise", "Ethan", "Farah")

)

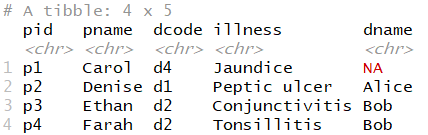
* 1. Use the data from the three tables given above to produce the following result. Note the order of the results for rows and columns should be the same as the output given below.



* 1. Write code using the appropriate tables given above to produce the following result:



* 1. Write code using appropriate tables given above to produce the following result:



* 1. What does the code *diagnosis %>% semi\_join(doctor)* produce? How many rows and columns are produced by this result?
  2. What does the code *diagnosis %>% anti\_join(doctor)* produce? How many rows and columns are produced by this result?
  3. Which of the two lines of code mentioned above (in questions 4d and 4e) displays the details of a diagnosis for which there is no matching doctor in the doctor table?

1. What is supervised learning different from unsupervised learning? Provide examples where these modes of learning might be useful.
2. What are the goals of regression, classification and clustering models? What are the types of problems these are best suited for?
3. Finally, here is a checklist. Do you know:
   1. How to use these functions: select, filter, mutate, arrange, group\_by and summarise?
   2. How to use gather, spread, unite and separate functions?
   3. How to scrape a table from an html page using functions in the rvest package?
   4. The difference between the four types of mutating joins (inner, left, right and full), and the filtering joins (semi and anti), and also when to use them?
   5. How to create models (a linear model and a random forest model) and also predict the output for a given new observation (or a set of observations)? Also, do you know how to compute R-squared values and interpret them? What does the residual plot tell us?
   6. How to read and write data using read\_csv() and write\_csv() functions?
   7. How to create visualisations using ggplot() as covered in the lectures, labs and mastery tasks?
   8. How to use the following functions: is.na(), ifelse() and cut()? Note: these functions were introduced in the first part of course. You’ll need to revisit appropriate course material for these functions.

**That’s it. All the best for the test!**