Project: Comparison of chronic disease prescriptions in major medical centers in Taiwan
By Jim

Introduction:

Taiwan's National Health Insurance was established in 1995 and has since become one of the most successful health insurance systems globally. The system's success can be attributed to the equal, convenient, and high-quality medical services it provides to all citizens, which in turn, helps alleviate the financial burden of medical expenses on families and individuals. Its low cost and convenience have made it internationally renowned and beneficial to its citizens. However, the system's affordability has led to an increase in medical consultations, which has resulted in a shortage of financial resources, leading to sustainability issues. The misuse of medical resources, heavy workload for medical staff, and short consultation times are among the consequences of this issue. Patients seeking medical treatment often face frequent medical consultations, repeated tests, and excessive prescription of medications.

Chronic disease-related consultations occupy a large amount of medical resources. In 2019, the medical expenses for the top ten diseases in Western medicine outpatient clinics (excluding emergency rooms) totaled 129.2 billion points, accounting for 32.4% of all Western medicine outpatient clinics (excluding emergency rooms). Observing the ranking of the top ten diseases with medical expenses, the first is chronic kidney disease, accounting for 12.9%; the second is type 2 diabetes, accounting for 5.3%; the third is essential hypertension, Accounting for 2.4%. (Ministry of Health and Welfare, 2019)

According to information released by Taiwan's Ministry of Health and Welfare, the most common age group using health insurance resources is the elderly over 65 years old reaching 38.3%. Owing to the longevity improvement, the proportion of the population of age over 65 has increased from 6.22% in 2001 to 10.63% in 2010. Studies show that the aging proportion will reach 24.37% in 2020 causing a huge cost to NHI, and suggests that the Government should be aware and prepared in advance. The data used are monthly data from 2000 to 2009, which is based on the unit of the month.(You-hsin Teng, 2011)

In summary, chronic diseases, especially the medical needs of the elderly population, are one of the important challenges facing NHI and need to be managed and addressed in a targeted manner.

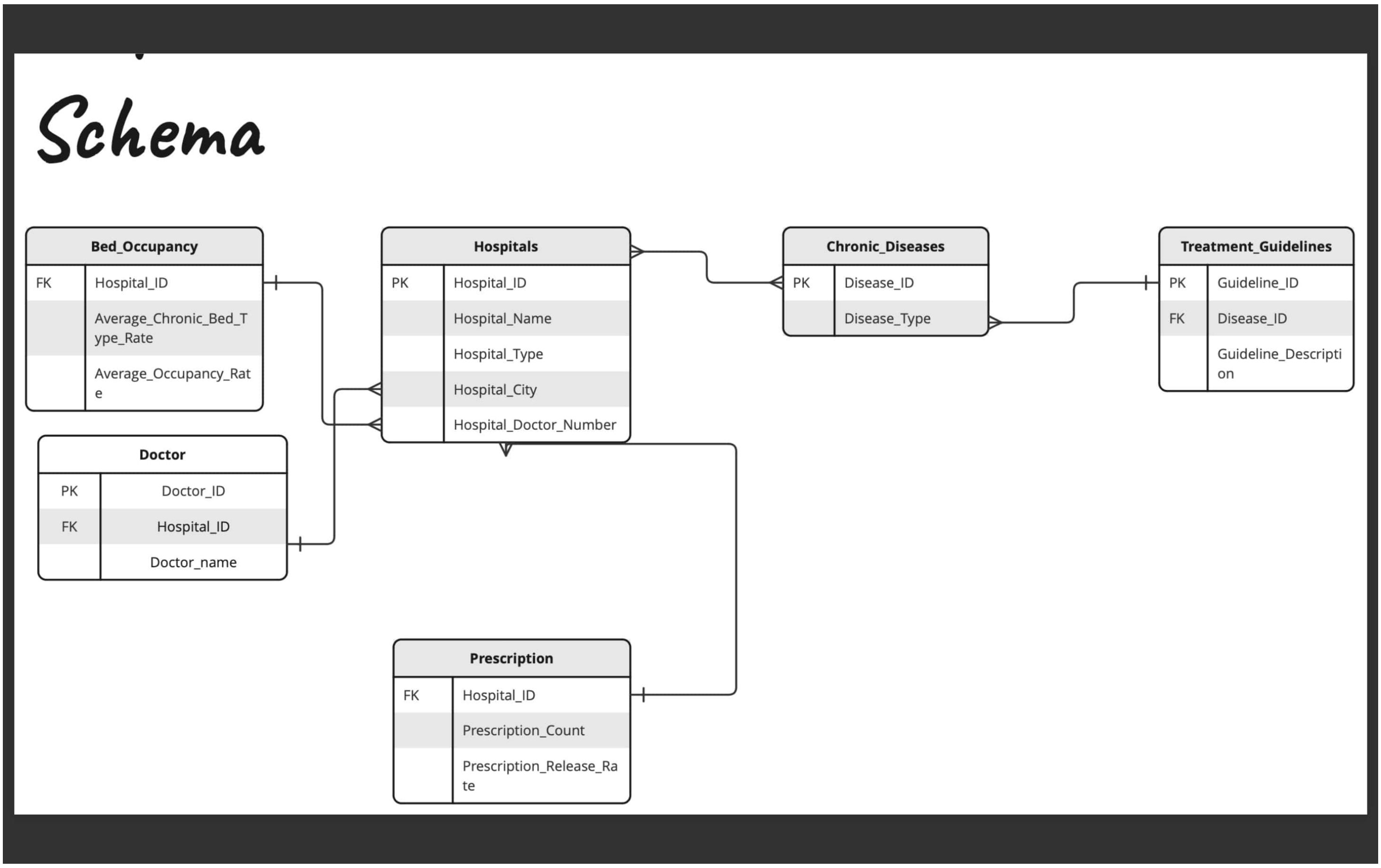
Program Description and Proposal:

This project adheres to the spirit of saving money and hopes to start by reducing unnecessary waste of medical resources. The purpose of this database is to assist healthcare providers, medical professionals, government agencies, and researchers in Taiwan. Its primary objective is to effectively manage and analyze data related to chronic disease management.

The database aims to eliminate the problem of scattered data by providing a centralized platform to track and analyze various aspects of chronic disease management, such as bed occupancy rates, prescription counts, prescription release rates, treatment guidelines, and medical resource quality indicators. The ultimate goal is to improve the quality of care for patients suffering from chronic diseases by enabling healthcare providers to make informed and data-driven decisions. The data sources for this database will be Data.gov.tw and the Department of Health database. The initial idea is that the user interface will include a form for inputting user information, a GPS map, and an introduction to chronic diseases. The goal is to allow users to enter their information, type of chronic disease, and location to quickly query the nearest local hospital with more medical resources.

Program Planning Process and Design:

This project uses Python pandas to organize part of the data, then uses DB browser to create a database, and finally uses R shiny to create a website and shinyapps.io to publish it. It is expected that 6 tables will be used. The schema used at the beginning is shown below:



Schema

The database under this plan will consist of the following 6 tables, including hospital, Doctor, prescription, Bed occupancy, Chronic disease, and Treatment Guideline. The hospital table mainly contains basic hospital information, such as hospital name, location, hospital type, and number of doctors. And use the hospital ID as the primary key. The hospital ID is mainly used to connect the prescription table and bed occupancy table. Then there is the doctor table, which has doctor-related information and doctor ID as the primary key, connecting chronic diseases, and chronic diseases are then connected to the treatment guideline. The chronic disease table contains 5 main chronic diseases hypertension, high_blood_sugar, hyperlipidemia, metabolic_syndrome, and chronic_kidney_disease. These five diseases account for approximately 70% of chronic diseases in Taiwan.

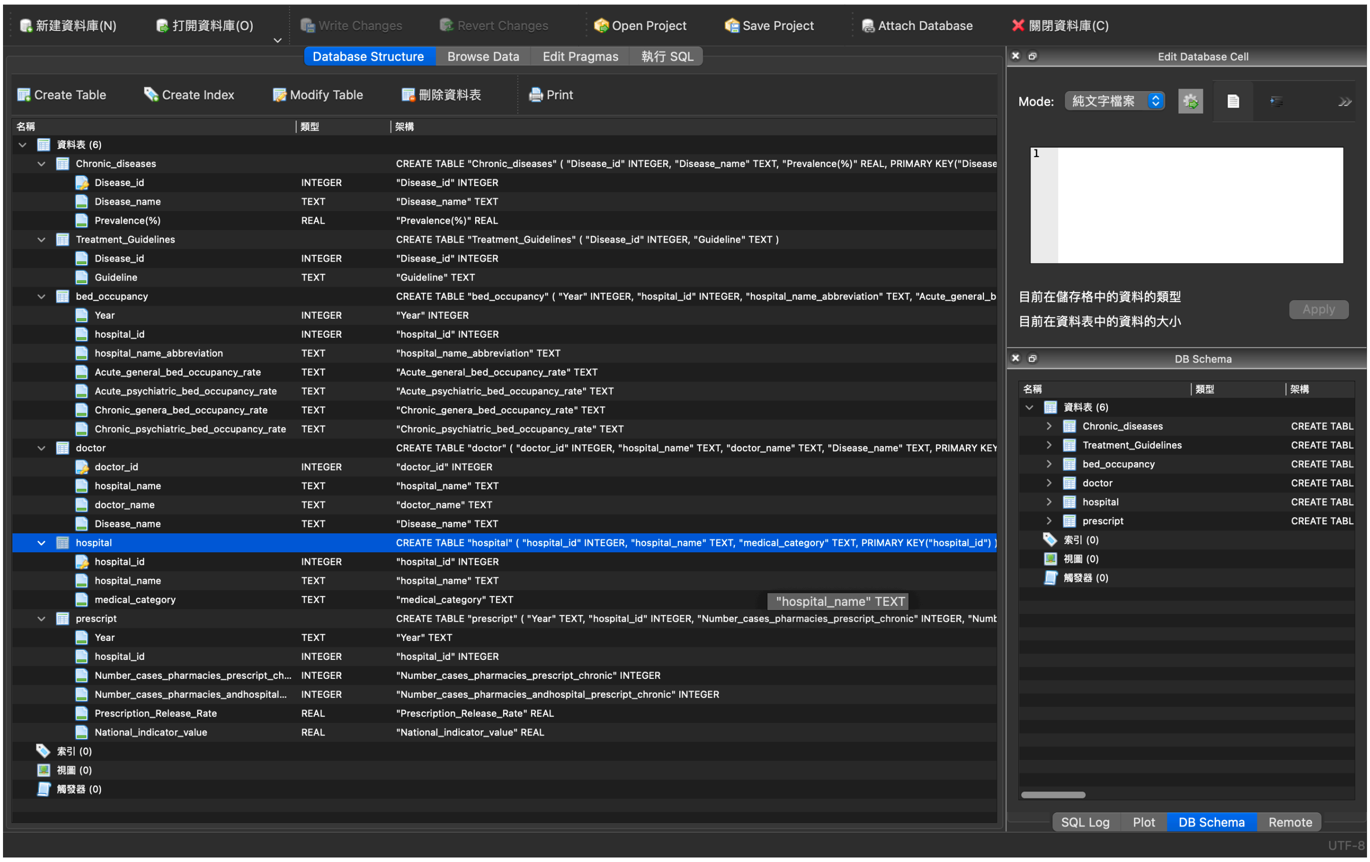
However, when I was creating the database, I encountered several problems. Firstly, I realized that the clinic data from the district that I had planned to use as the primary information source was insufficient. To get comprehensive prescription data, I had to narrow down the information source to medical centers. This limitation can affect the completeness of the database, and it might limit the scope of analysis, especially for healthcare services in smaller clinics or community health centers. It also led to the infeasibility of GIS map production. An overly large range of distance analysis loses the original meaning of making a GIS map to analyze the hospitals closest to the location.

Secondly, organizing doctor information was a challenge. Especially when trying to use Python for web crawling, each hospital saves data in a different format. As a result, there were inconsistencies and difficulties in integrating the data. I had to create a portion of doctor information manually, which might lead to inaccuracies and potential data gaps. The presence of doctors with multiple and repeated specialties also added complexity, which raised concerns about data normalization and effective data utilization.

Thirdly, limitations in the schema design became apparent during the production phase. For instance, the inclusion of hospital_id between the first four tables and appending the hospital_name column after each table resulted in redundant and uninformative data. This structure lacks practicality and hinders efficient data retrieval and analysis, highlighting the need for a more streamlined and effective schema design.

Lastly, I planned to incorporate images into the database, linked to the hospital or doctor tables but did not complete it. Although images can enhance data presentation and user experience, the incomplete implementation of this feature reflects a missed opportunity to enrich the database's functionality and visual appeal.

DB Browser:



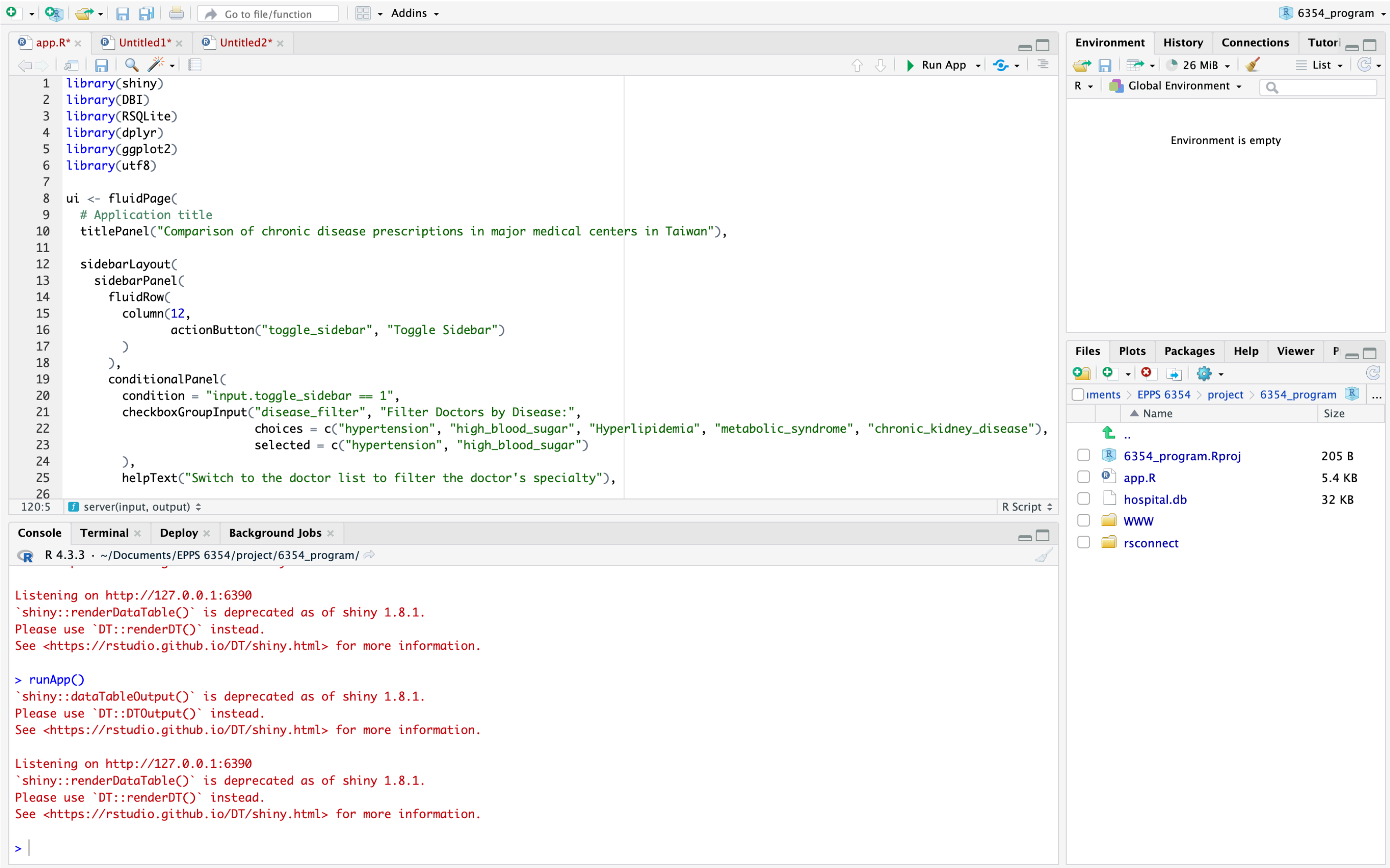
Overall database

Database tables

	doctor_id	hospital_name	doctor_name	Disease_name
	過濾	過濾	過濾	過濾
1	601	國立臺灣大學醫學院附設醫院	YI-CHIH WANG	hypertension
2	602	國立臺灣大學醫學院附設醫院	Tzung-Dau Wang	hypertension
3	603	國立臺灣大學醫學院附設醫院	Chien-Hsieh Chiang	high_blood_sugar
4	604	國立臺灣大學醫學院附設醫院	Chih-Yuan Wang	Hyperlipidemia
5	605	國立臺灣大學醫學院附設醫院	Li-Tan Yang	Hyperlipidemia
6	606	國立臺灣大學醫學院附設醫院	Chih-Yuan Wang	metabolic_syndrome
7	607	國立臺灣大學醫學院附設醫院	Lee Chin Wong	metabolic_syndrome
8	608	國立臺灣大學醫學院附設醫院	Chia-Ter Chao	chronic_kidney_disease
9	609	國立臺灣大學醫學院附設醫院	TAI-SHUAN LAI	chronic_kidney_disease
10	701	三軍總醫院附設民眾診療服務處	Yi-Jen Hung	hypertension
11	702	三軍總醫院附設民眾診療服務處	Su Shengqiang	high_blood_sugar
12	703	三軍總醫院附設民眾診療服務處	Guo Fuzhi	Hyperlipidemia
13	704	三軍總醫院附設民眾診療服務處	Chieh-Hua Lu	metabolic_syndrome
14	705	三軍總醫院附設民眾診療服務處	Chih-Chien Sung	chronic_kidney_disease
15	801	臺北榮民總醫院	YAW ZOW DIN	hypertension
16	802	臺北榮民總醫院	Kang-Ling Wang	hypertension
17	803	臺北榮民總醫院	Yi-Jen Wang	hypertension
18	804	臺北榮民總醫院	Wei-Ting Wang	hypertension
19	805	臺北榮民總醫院	Chern-En Chiang	hypertension
20	806	臺北榮民總醫院	Wen-Chung Yu	hypertension
21	807	臺北榮民總醫院	Cheng-Hsueh Wu	hypertension
22	808	臺北榮民總醫院	Tsai-Hung Wu	hypertension
23	809	臺北榮民總醫院	Cheng-I Wu	hypertension
24	810	臺北榮民總醫院	Tao-Cheng Wu	hypertension
25	811	臺北榮民總醫院	Hsin-Bang Leu	hypertension
26	812	臺北榮民總醫院	Shih-Hsien Sung	hypertension
27	813	臺北榮民總醫院	Ching-Wei Lee	hypertension
28	814	臺北榮民總醫院	Chia-Yu Chou	hypertension
29	815	臺北榮民總醫院	Y. C. LIN	high_blood_sugar
30	816	臺北榮民總醫院	Liang-Yu Lin	high_blood_sugar

Doctor table overview

R programming Shiny:



Shiny overview

Several special packages are used including The DBI and RSQLite packages. They are important tools in R for working with databases. The DBI Package (Database Interface) provides a consistent and unified way to interact with different database systems in R. It offers functions and methods for connecting to databases, executing SQL queries, retrieving data, and managing database transactions. With DBI, you can work with various database backends like SQLite, MySQL, PostgreSQL, Oracle, and more, using a standardized set of functions. This package simplifies database operations in R by abstracting the differences between database systems and providing a common framework for database interactions.

The RSQLite package is a specific database backend for SQLite databases within the DBI framework. It allows you to create, read, update, and delete data in SQLite databases directly from R. SQLite is a lightweight, serverless, self-contained database engine that is widely used for embedded databases and small to medium-sized applications. RSQLite provides functions and methods that integrate seamlessly with DBI, enabling R users to work with SQLite databases using familiar DBI syntax and conventions.

In addition, since Traditional Chinese is used in the database. The utf8 package in R is used for handling UTF-8 encoded text data.

In the user interface (UI), there is a toggle sidebar that can be used to hide certain parts of the layout to make it look neater. Once clicked, it will reveal a checkbox to filter chronic disease types, a slider to adjust the amount of data displayed, and a select table to choose the table you want to view. Then a data table was placed in the mainPanel part to display the table data. A chart made with a ggplot2 was used to observe the usage of patent medicines in the hospital.

```r
library(shiny)
library(DBI)
library(RSQLite)
library(dplyr)
library(ggplot2)
library(utf8)

ui <- fluidPage(
  # Application title
  titlePanel("Comparison of chronic disease prescriptions in major
             medical centers in Taiwan"),

  sidebarLayout(
    sidebarPanel(
      fluidRow(
        column(12,
               actionButton("toggle_sidebar", "Toggle Sidebar")
        )
      ),
      conditionalPanel(
        condition = "input.toggle_sidebar == 1",
        checkboxGroupInput("disease_filter", "Filter Doctors by Disease:",
                           choices = c("hypertension", "high_blood_sugar",
                                       "Hyperlipidemia", "metabolic_syndrome",
                                       "chronic_kidney_disease"),
                           selected = c("hypertension", "high_blood_sugar")
        ),
        helpText("Switch to the doctor list to filter the doctor's specialty"),

        sliderInput("nrows", "Enter the number of rows to display:",
                    min = 1,
                    max = 100,
                    value = 10),
        selectInput("table_select", "Select table:",
                    choices = c("hospital", "doctor", "prescript",
                                "bed_occupancy", "Chronic_diseases",
                                "Treatment_Guidelines"),
                    selected = "hospital")
      )
    ),

    mainPanel(
      fluidRow(
        column(12,
               dataTableOutput("tbl"),
               plotOutput("plot")
        )
      ),
      img(src = "schema.jpg", height = 680, width = 1200)
    )
  )
)

# Define server logic
```

In the Server part, Establish a connection with the hospital database through the line: ‘sqlite_conn <- dbConnect(RSQLite::SQLite(), “hospital.db”)'

‘query <- paste0("SELECT d.doctor_id, d.hospital_name, d.doctor_name, d.Disease_name, h.hospital_id FROM doctor d LEFT JOIN hospital h ON d.hospital_name = h.hospital_name WHERE d.Disease_name IN (", selected_diseases_str , ") LIMIT ", input$nrows)’

This code is used to create an SQL query. The query's purpose is to select data in a specific field from the data table named "doctor", and at the same time perform a left join (LEFT JOIN) to the data table named "hospital". It also filters out matching data columns based on specific conditions.

Use the paste0 function to concatenate multiple strings together to create a SQL query statement.

"SELECT d.doctor_id, d.hospital_name, d.doctor_name, d.Disease_name, h.hospital_id": Selects fields, including the doctor_id, hospital_name, doctor_name, and Disease_name fields in the "doctor" data table(d), and the "hospital" data table's(h) hospital_id field.

"FROM doctor d LEFT JOIN hospital h ON d.hospital_name = h.hospital_name": Specifies the source and connection method of the data table. LEFT JOIN means retaining all the data columns in the "doctor" data table and adding the data in the "hospital" data table that meets the conditions. The join condition is based on the hospital_name column of the "doctor" data table being equal to the hospital_name column of the "hospital" data table.

"WHERE d.Disease_name IN (", selected_diseases_str, ")": Sets filter conditions to select only data that matches the specified disease name. selected_diseases_str is a string containing a comma-separated list of disease names that will be used for filtering.

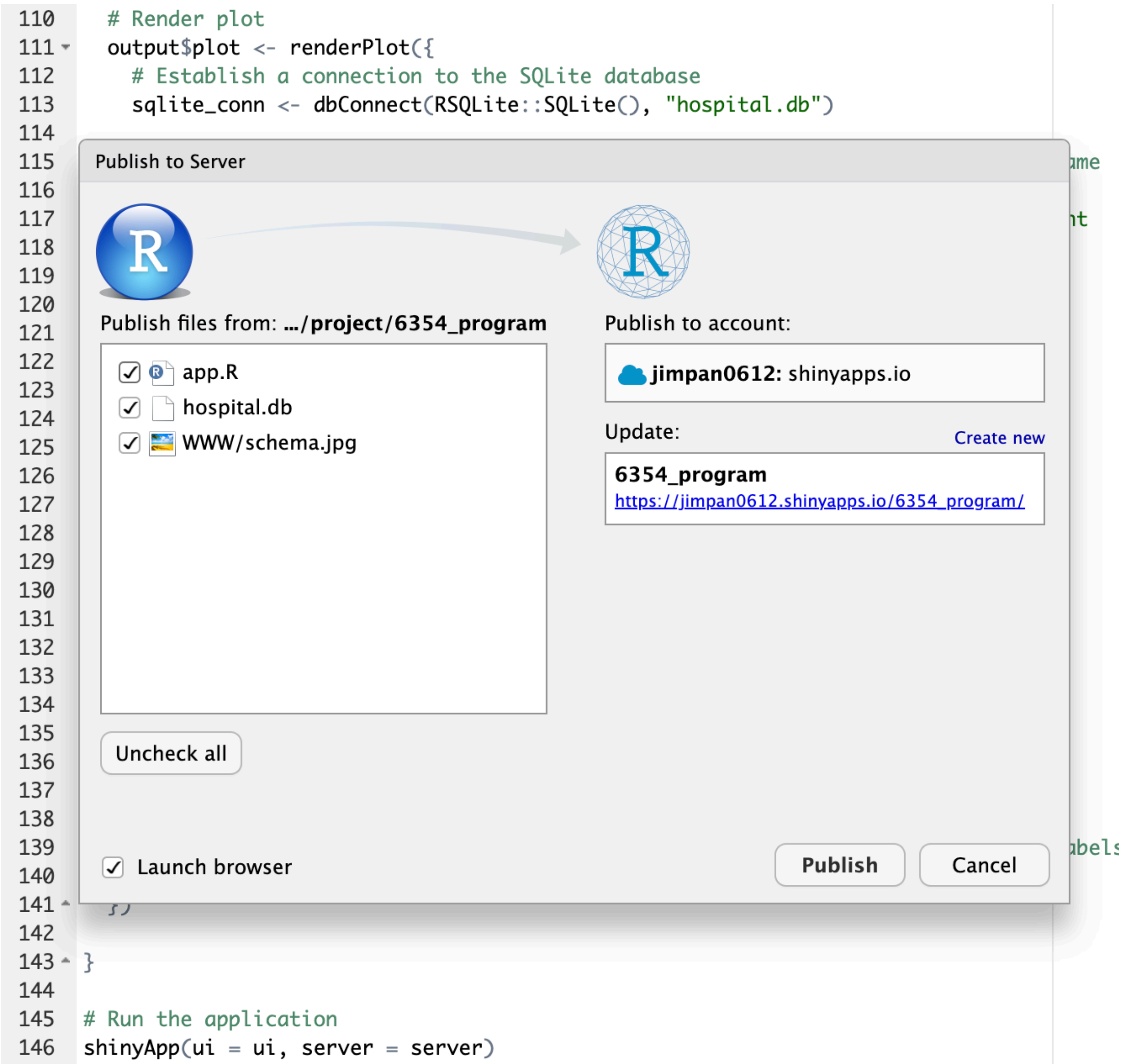
"LIMIT ", input$nrows: Limits the number of columns in query results. input$nrows is a variable used to specify the maximum number of data columns to be returned.

```r
# Define server logic
server <- function(input, output) {
  output$tbl <- renderDataTable({
    # Establish a connection to the SQLite database
    sqlite_conn <- dbConnect(RSQLite::SQLite(), "hospital.db")

    # Create SQL query to retrieve data from the selected table with JOIN
    if (input$table_select == "doctor") {
      # Filter doctors based on selected diseases
      selected_diseases <- input$disease_filter
      if (length(selected_diseases) > 0) {
        # Construct the IN clause for Disease_name
        selected_diseases_str <- paste0("'", selected_diseases, "'",
                                        collapse = ", ")
        query <- paste0("SELECT d.doctor_id, d.hospital_name, d.doctor_name,
        d.Disease_name, h.hospital_id
                         FROM doctor d
                         LEFT JOIN hospital h ON d.hospital_name = h.hospital_name
                         WHERE d.Disease_name IN (", selected_diseases_str, ")
                         LIMIT ", input$nrows)
      } else {
        query <- paste0("SELECT d.doctor_id, d.hospital_name, d.doctor_name,
        d.Disease_name, h.hospital_id
                         FROM doctor d
                         LEFT JOIN hospital h ON d.hospital_name = h.hospital_name
                         LIMIT ", input$nrows)
      }
    } else if (input$table_select == "prescript") {
      query <- paste0("SELECT p.*, h.hospital_name
                       FROM prescript p
                       LEFT JOIN hospital h ON p.hospital_id = h.hospital_id
                       LIMIT ", input$nrows)
    } else if (input$table_select == "bed_occupancy") {
      query <- paste0("SELECT b.*, h.hospital_name
                       FROM bed_occupancy b
                       LEFT JOIN hospital h ON b.hospital_id = h.hospital_id
                       LIMIT ", input$nrows)
    } else if (input$table_select == "Chronic_diseases") {
      query <- paste0("SELECT * FROM Chronic_diseases LIMIT ", input$nrows)
    } else if (input$table_select == "Treatment_Guidelines") {
      query <- paste0("SELECT * FROM Treatment_Guidelines LIMIT ", input$nrows)
    } else {
      query <- paste0("SELECT * FROM ", input$table_select, " LIMIT ", input$nrows)
    }

    # Execute SQL query to retrieve data
    data <- dbGetQuery(sqlite_conn, query)

    # Close the database connection
    dbDisconnect(sqlite_conn)

    # Return the data to be displayed in the data table
    data
  })

```

This last paragraph uses ggplot2 to draw the chart. Use JOIN to connect the hospital table and prescription table to observe which hospitals produce the most prescription drugs.

```
    # Render plot
    output$plot <- renderPlot({
      # Establish a connection to the SQLite database
      sqlite_conn <- dbConnect(RSQLite::SQLite(), "hospital.db")

      # Create SQL query to calculate prescript count by hospital_id and hospital_name
      prescript_count_query <- "SELECT p.hospital_id, h.hospital_name,
      SUM(p.Number_cases_pharmacies_andhospital_prescript_chronic) AS prescript_count
                              FROM prescript p
                              LEFT JOIN hospital h ON p.hospital_id = h.hospital_id
                              GROUP BY p.hospital_id, h.hospital_name
                              ORDER BY prescript_count DESC
                              LIMIT 10"

      # Execute SQL query to retrieve prescript count data
      prescript_count_data <- dbGetQuery(sqlite_conn, prescript_count_query)

      # Close the database connection
      dbDisconnect(sqlite_conn)

      # Plot the prescript count data with adjusted x and y axis
      ggplot(prescript_count_data, aes(x = reorder(hospital_name, prescript_count),
                                       y = prescript_count)) +
        geom_bar(stat = "identity", fill = 'orange', color = 'cyan') +
        labs(title = "Top 10 Hospitals by Prescript Count", x = "Hospital Name",
             y = "Prescript Count") +
        scale_x_discrete(labels = function(x) utf8::utf8_encode(x)) +
        # Encode labels in UTF-8 for traditional Chinese characters
        scale_y_continuous(labels = scales::comma) +  # Adjust y axis labels format
        theme(axis.text.x = element_text(angle = 45, hjust = 1))  # Rotate x axis labels

    })

  }

  # Run the application
  shinyApp(ui = ui, server = server)
```

Publish to Shinyapps.io



Result:

Id	Name	Status	Instances	Deployed Date	Created Date
11923243	6354_program	Running	1	May 2, 2024	May 2, 2024

6354 jimpan0612.shinyapps.io

Comparison of chronic disease prescriptions in major medical centers in Taiwan

Toggle Sidebar

Show 25 entries

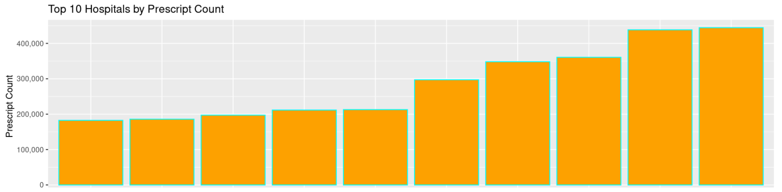
Search:

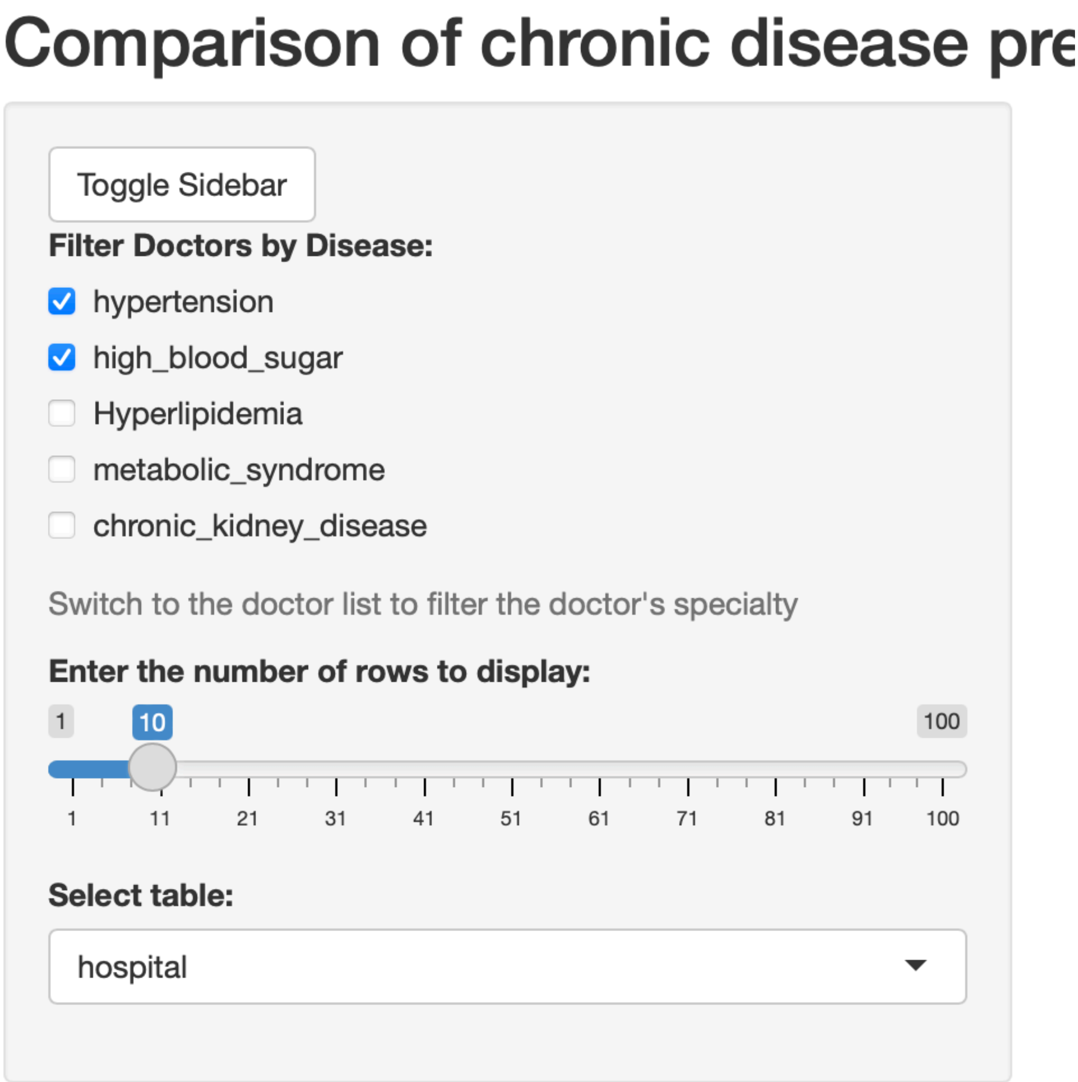
hospital_id	hospital_name	medical_category
602030026	高雄榮民總醫院	醫學中心
501110514	三軍總醫院附設民眾診療服務處	醫學中心
1101150011	新光醫療財團法人新光吳火獅紀念醫院	醫學中心
421040011	國立成功大學醫學院附設醫院	醫學中心
601160016	臺北榮民總醫院	醫學中心
401180014	國立臺灣大學醫學院附設醫院	醫學中心
1141310019	奇美醫療財團法人奇美醫院	醫學中心
617060018	臺中榮民總醫院	醫學中心
1317050017	中國醫藥大學附設醫院	醫學中心
1131010011	醫療財團法人徐元智先生醫藥基金會亞東紀念醫院	醫學中心
hospital_id	hospital_name	medical_category

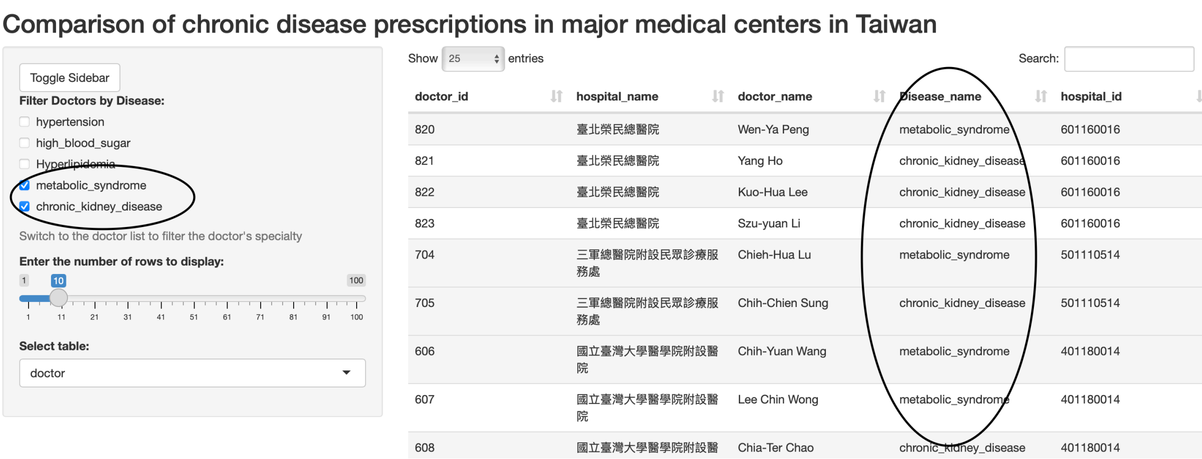
Showing 1 to 10 of 10 entries

Previous 1 Next

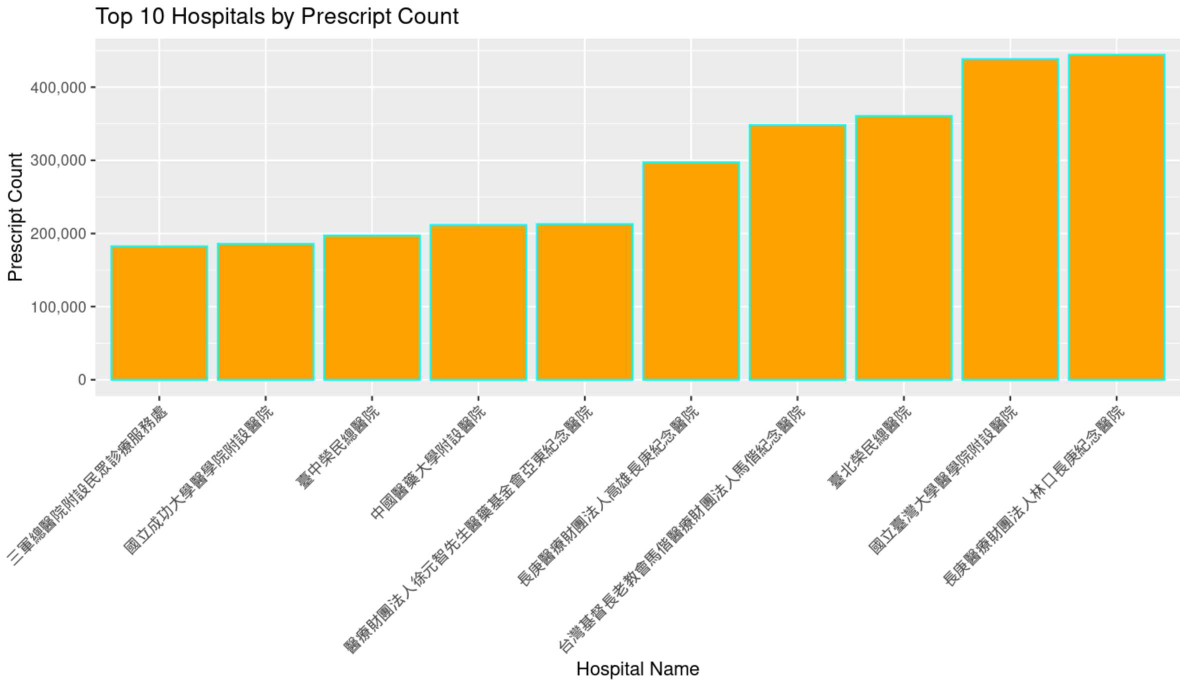
Top 10 Hospitals by Prescript Count







doctor_id	hospital_name	doctor_name	Disease_name	hospital_id
820	臺北榮民總醫院	Wen-Ya Peng	metabolic_syndrome	601160016
821	臺北榮民總醫院	Yang Ho	chronic_kidney_disease	601160016
822	臺北榮民總醫院	Kuo-Hua Lee	chronic_kidney_disease	601160016
823	臺北榮民總醫院	Szu-yuan Li	chronic_kidney_disease	601160016
704	三軍總醫院附設民眾診療服務處	Chieh-Hua Lu	metabolic_syndrome	501110514
705	三軍總醫院附設民眾診療服務處	Chih-Chien Sung	chronic_kidney_disease	501110514
606	國立臺灣大學醫學院附設醫院	Chih-Yuan Wang	metabolic_syndrome	401180014
607	國立臺灣大學醫學院附設醫院	Lee Chin Wong	metabolic_syndrome	401180014
608	國立臺灣大學醫學院附設醫	Chia-Ter Chao	chronic_kidney_disease	401180014



Conclusion:

In conclusion, The analysis highlights that chronic diseases have a significant impact on healthcare resources, highlighting the need for better data management solutions. The proposed centralized database project is a proactive approach to improving chronic disease management. Despite challenges like data source limitations and inconsistencies in doctor information, using technologies like Python, DB Browser, and R programming with specialized packages like DBI and RSQLite shows a commitment to data-driven solutions. Moving forward, it is crucial to prioritize interface improvements such as GIS mapping for localized healthcare services, optimize data retrieval processes, and refine schema design. These enhancements will not only improve the user experience but also facilitate efficient data analysis and decision-making for healthcare providers and policymakers.

Reference:

You-hsin Teng. 2011. “Second-generation cash flow test of national health insurance: a bankruptcy model of an aging population”

Ministry of Health and Welfare. 2019. “National HealthInsurance Medical Statistics”

Ministry of Health and Welfare. 2017. “National Health Interview Survey”