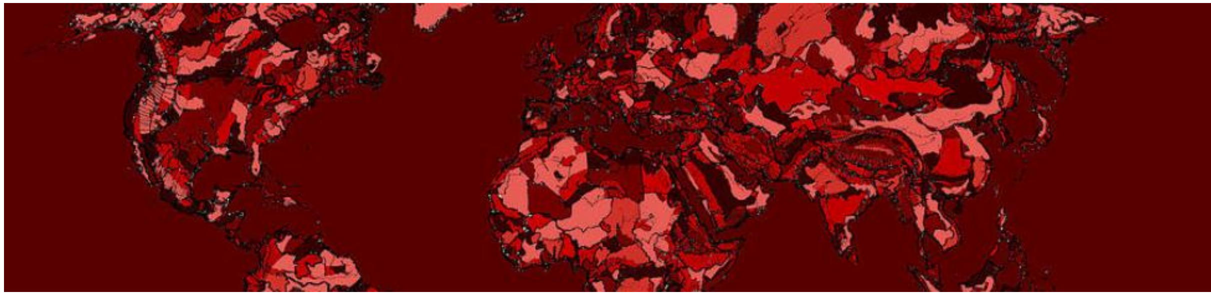
COM506 Term Project
Creating Freight Indexes Information

Hyuncheol Ryu

August 2022

Swiss Institute of
Artificial Intelligence



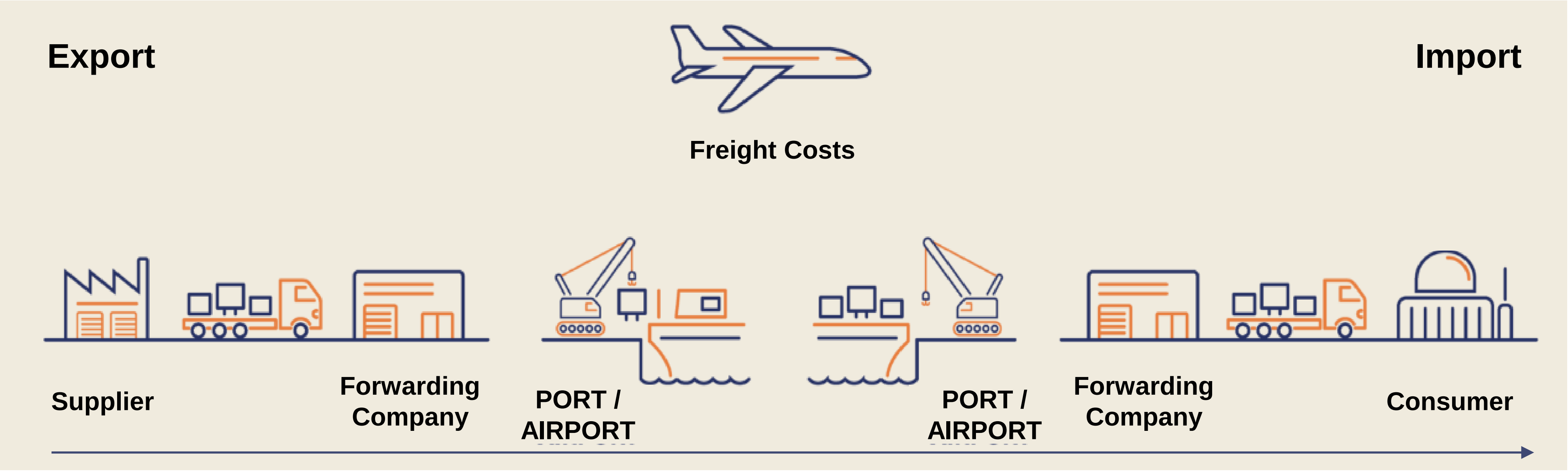
Contents

Section

Forwarding and Pricing

- Forwarding is a business that arranges the logistics of import and export cargo. Pricing, one of the forwarding tasks, plays the role of a negotiator by purchasing supplies from airlines and shipping companies at low prices and charging customers for logistics costs at high prices.

Freight Flow



(https://osprey.group/our-expertise/project-freight-forwarding/)

- Forwarding refers to the business of arranging the logistics of import/export cargo by using the logistics facilities and equipment of the supplier in the name of the forwarding company at the request of the consumer. **The consumer will want to charge the forwarding company for freight costs at the lowest possible price, and the supplier will want to collect the logistics and transportation costs at the highest possible price.**

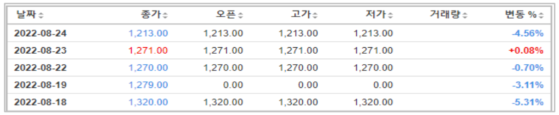
- **At this point, the need for the forwarding company's pricing task arises.** That is, the person in charge of pricing is responsible for establishing and implementing strategies for air and ocean international freight rates, contacting airlines and shipping companies to identify trends, and securing supply space through seasonal cargo capacity management.

Issues in Pricing tasks

- Due to the nature of the pricing tasks, it is necessary to check the freight indexes daily. However, as the management subject for each freight index is different, work inefficiency occurs in the pricing process. To improve this, it is necessary to make it possible to view and manage the entire index in one channel.

Freight Indexes List

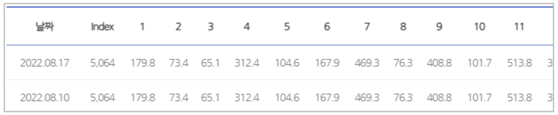
BDI (Baltic Dry Index)



날짜	종가	오픈	고가	저가	거래량	변동 %
2022-08-24	1,213.00	1,213.00	1,213.00	1,213.00		-4.56%
2022-08-23	1,271.00	1,271.00	1,271.00	1,271.00		+0.08%
2022-08-22	1,270.00	1,270.00	1,270.00	1,270.00		-0.70%
2022-08-19	1,279.00	0.00	0.00	0.00		-3.11%
2022-08-18	1,320.00	1,320.00	1,320.00	1,320.00		-5.31%

(https://kr.investing.com/indices/baltic-dry-historical-data)

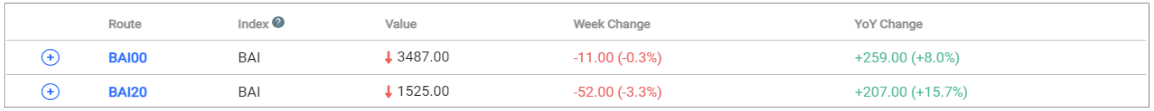
HRCI (Howe Robinson Container Index)



날짜	Index	1	2	3	4	5	6	7	8	9	10	11	
2022.08.17	5,064	179.8	73.4	65.1	312.4	104.6	167.9	469.3	76.3	408.8	101.7	513.8	3
2022.08.10	5,064	179.8	73.4	65.1	312.4	104.6	167.9	469.3	76.3	408.8	101.7	513.8	3

(https://www.ksg.co.kr/shippingGraph/hrci_graph.jsp)

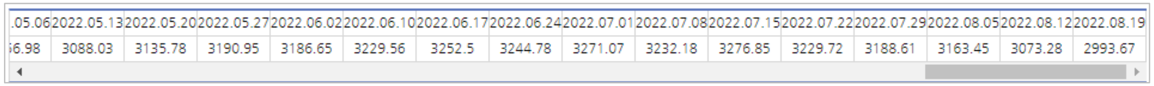
BAI (Baltic Air Freight Indices)



	Route	Index	Value	Week Change	YoY Change
⊕	BAI00	BAI	↓ 3487.00	-11.00 (-0.3%)	+259.00 (+8.0%)
⊕	BAI20	BAI	↓ 1525.00	-52.00 (-3.3%)	+207.00 (+15.7%)

(https://dashboard.tacindex.com/dashboard)

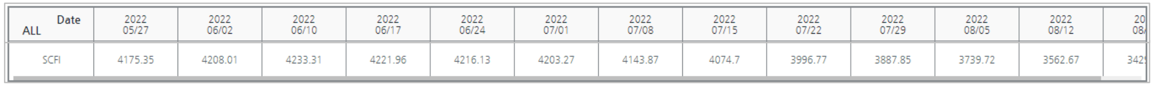
CCFI (China Containerized Freight Index)



.05.06	2022.05.13	2022.05.20	2022.05.27	2022.06.02	2022.06.10	2022.06.17	2022.06.24	2022.07.01	2022.07.08	2022.07.15	2022.07.22	2022.07.29	2022.08.05	2022.08.12	2022.08.19
6.98	3088.03	3135.78	3190.95	3186.65	3229.56	3252.5	3244.78	3271.07	3232.18	3276.85	3229.72	3188.61	3163.45	3073.28	2993.67

(https://www.kcla.kr/web/inc/html/4-1_2.asp)

SCFI (Shanghai Containerized Freight Index)



Date ALL	2022 05/27	2022 06/02	2022 06/10	2022 06/17	2022 06/24	2022 07/01	2022 07/08	2022 07/15	2022 07/22	2022 07/29	2022 08/05	2022 08/12	20 08/
SCFI	4175.35	4208.01	4233.31	4221.96	4216.13	4203.27	4143.87	4074.7	3996.77	3887.85	3739.72	3562.67	342

(https://www.tradlinx.com/freight-index)

- There are a total of 5 freight indexes that should be referenced in the pricing tasks
- Each index has a different management entity, so the person in charge must visit a different web page to check the index data

Key Findings

Inefficiency in the pricing process

- Time wasted in the process of visiting each index web page and checking the index
- Data input processing and accumulation management for daily index data is done manually

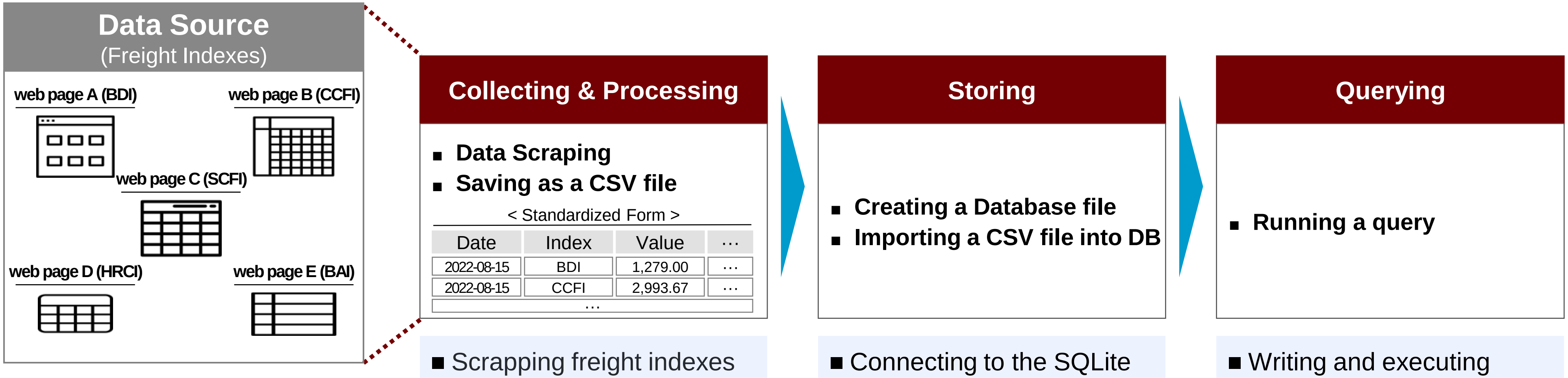
Need Improvement

- **It should be possible to check all indexes in one channel**
- **Daily freight indexes value must be stored in a database**

Implementation Plan

- The team established an implementation plan for the Collecting-Processing-Storing-Querying procedure so that freight indexes can be viewed and managed in one channel.

Problem Solving Process



Collecting & Processing

- **Data Scraping**
- **Saving as a CSV file**

< Standardized Form >

Date	Index	Value	…
2022-08-15	BDI	1,279.00	…
2022-08-15	CCFI	2,993.67	…
…			

Storing

- **Creating a Database file**
- **Importing a CSV file into DB**

Querying

- **Running a query**

Data Source

- Defining the data source that we want to scrap
- Designing the ERD of the database

Collecting & Processing

- Scrapping freight indexes for each web page using BeautifulSoup, Selenium packages
- Transforming table of indexes data into a standardized form
- Saving it as a CSV files

Storing

- Connecting to the SQLite DB file and importing the CSV files through Python
- Assigning a primary key and a foreign key to each table

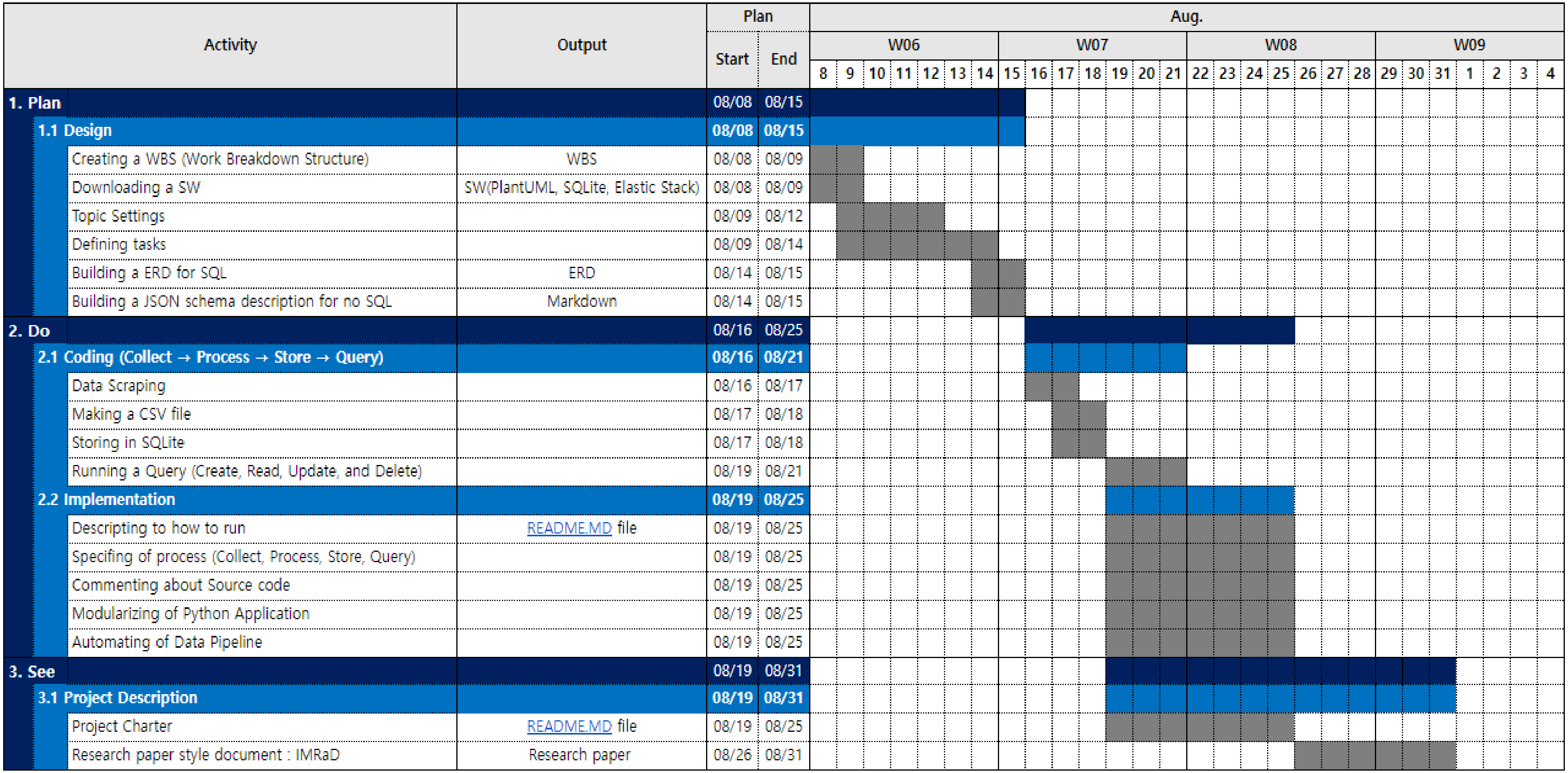
Querying

- Writing and executing query statements such as Create, Read, Update, and Delete

Project Schedule

- The project was carried out over a total of 24 days from 8/8 to 8/31.

Schedule Chart



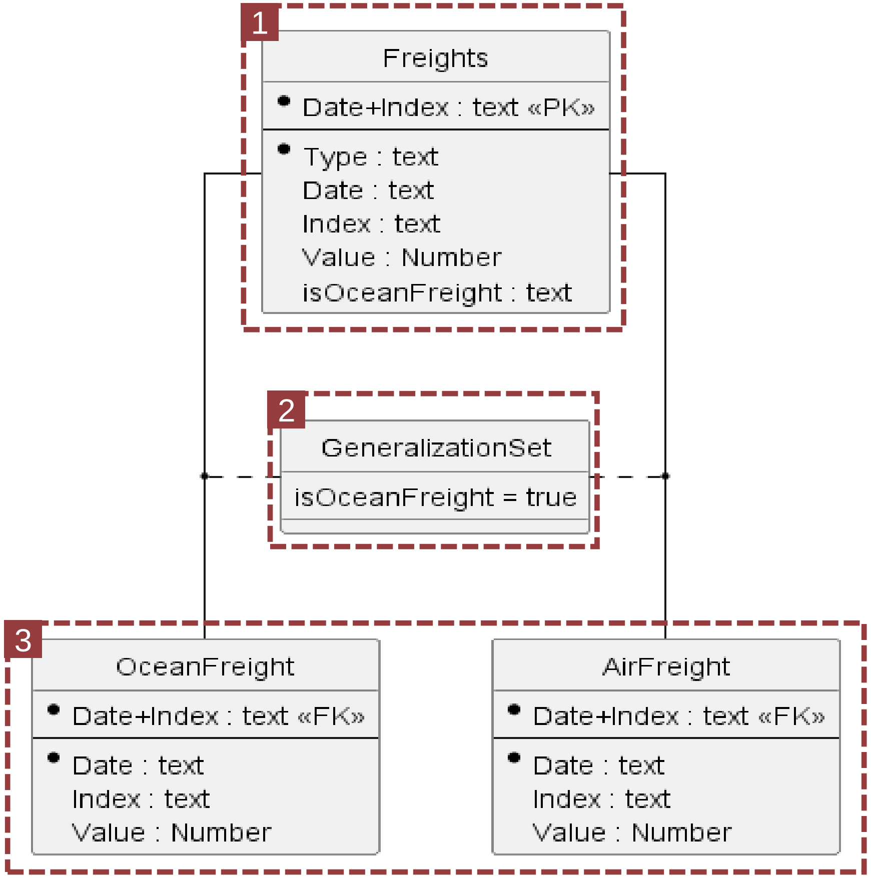
Activity	Output	Plan		Aug.																											
		Start	End	W06							W07							W08							W09						
				8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	1	2	3	4
1. Plan		08/08	08/15	■	■	■	■	■	■	■	■																				
1.1 Design		**08/08**	**08/15**	■	■	■	■	■	■	■	■																				
Creating a WBS (Work Breakdown Structure)	WBS	08/08	08/09	■	■																										
Downloading a SW	SW(PlantUML, SQLite, Elastic Stack)	08/08	08/09	■	■																										
Topic Settings		08/09	08/12		■	■	■	■																							
Defining tasks		08/09	08/14		■	■	■	■	■	■																					
Building a ERD for SQL	ERD	08/14	08/15							■	■																				
Building a JSON schema description for no SQL	Markdown	08/14	08/15							■	■																				
2. Do		08/16	08/25									■	■	■	■	■	■	■	■	■	■										
2.1 Coding (Collect → Process → Store → Query)		**08/16**	**08/21**									■	■	■	■	■	■														
Data Scraping		08/16	08/17									■	■																		
Making a CSV file		08/17	08/18										■	■																	
Storing in SQLite		08/17	08/18										■	■																	
Running a Query (Create, Read, Update, and Delete)		08/19	08/21												■	■	■														
2.2 Implementation		**08/19**	**08/25**												■	■	■	■	■	■	■										
Descripting to how to run	README.MD file	08/19	08/25												■	■	■	■	■	■	■										
Specifing of process (Collect, Process, Store, Query)		08/19	08/25												■	■	■	■	■	■	■										
Commenting about Source code		08/19	08/25												■	■	■	■	■	■	■										
Modularizing of Python Application		08/19	08/25												■	■	■	■	■	■	■										
Automating of Data Pipeline		08/19	08/25												■	■	■	■	■	■	■										
3. See		08/19	08/31												■	■	■	■	■	■	■	■	■	■	■	■	■				
3.1 Project Description		**08/19**	**08/31**												■	■	■	■	■	■	■	■	■	■	■	■	■				
Project Charter	README.MD file	08/19	08/25												■	■	■	■	■	■	■										
Research paper style document : IMRaD	Research paper	08/26	08/31																			■	■	■	■	■	■				

Entity Relationship Diagram (ERD)

- The team designed the ERD which is a database design for the supertype/subtype relationship using PlantUML.

Description of ERD

Relationship Between Supertype & Subtype Entities



1

Freights

- Date+Index : text «PK»

- Type : text
Date : text
Index : text
Value : Number
isOceanFreight : text

2

GeneralizationSet

isOceanFreight = true

3

OceanFreight

- Date+Index : text «FK»

- Date : text
Index : text
Value : Number

AirFreight

- Date+Index : text «FK»

- Date : text
Index : text
Value : Number

Description

1 **Supertype Entity : Freights**

- **Attributes**
  - – Type
    - – One of 'OceanFreight' and 'AirFreight'
  - – Date
    - – The date the value was recorded
  - – Index
    - – One of 'BDI' & 'CCFI' & 'SCFI' & 'HRCI' & 'BAI'
  - – Value
    - – Index Value
  - – IsOceanFreight
    - – 'True' if Type is OceanFreight
  - – Date+Index (PK)
    - – A Combination of Date and Index, Which is the PK of this table

2 **GeneralizationSet**

- If the value in the IsOceanFreight is true, the OceanFreight table; otherwise, the AirFreight table

3 **Supertype Entities : OceanFrieght, AirFreight**

- Other than the Date+Index designated as FK, the remaining columns are the same as the Freights entity

Collecting & Processing

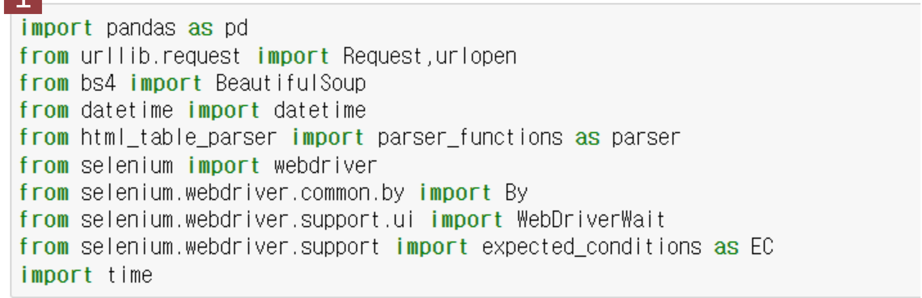
Collecting ▸ Processing ▸ Storing ▸ Querying

- A total of 7 packages were used to scrape and process index data.

Description of 'Collecting & Processing' Process

Code

1



```
import pandas as pd
from urllib.request import Request,urlopen
from bs4 import BeautifulSoup
from datetime import datetime
from html_table_parser import parser_functions as parser
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC
import time
```

Description

1 **Importing the necessary packages**

■ **Packages**

– Pandas
  – It is used to convert index data scraped by web pages into a data frame.
– Request, urlopen
  – It is a package required to access URLs with index information.
– BeautifulSoup
  – It is a Python package for parsing HTML and XML documents.
– Datetime
  – It is a package to put a time value in columns that do not have a Date value.
– Html_table_parser
  – It is a package that parses table in HTML.
– Selenium
  – It is a package needed to scrape data created dynamically by javascript.
– Time
  – It is a package required to wait for the loading of dynamic web pages.

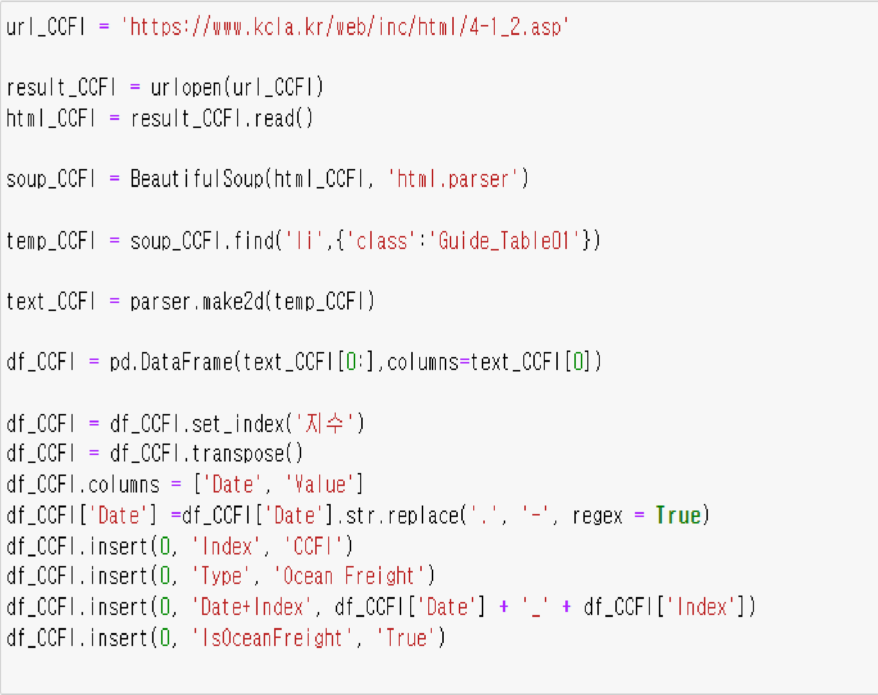
Collecting & Processing

Collecting ▸ Processing ▸ Storing ▸ Querying

- The team scraped the index data from the web page for each index, changed it into a data frame, and matched it to a standardized format.

Description of 'Collecting & Processing' Process

Code



```
1 url_CCFI = 'https://www.kcla.kr/web/inc/html/4-1_2.asp'

2 result_CCFI = urlopen(url_CCFI)
  html_CCFI = result_CCFI.read()

3 soup_CCFI = BeautifulSoup(html_CCFI, 'html.parser')

4 temp_CCFI = soup_CCFI.find('li',{'class':'Guide_Table01'})

5 text_CCFI = parser.make2d(temp_CCFI)

6 df_CCFI = pd.DataFrame(text_CCFI[0:],columns=text_CCFI[0])

7 df_CCFI = df_CCFI.set_index('지수')
  df_CCFI = df_CCFI.transpose()
  df_CCFI.columns = ['Date', 'Value']
  df_CCFI['Date'] =df_CCFI['Date'].str.replace('.', '-', regex = True)
  df_CCFI.insert(0, 'Index', 'CCFI')
  df_CCFI.insert(0, 'Type', 'Ocean Freight')
  df_CCFI.insert(0, 'Date+Index', df_CCFI['Date'] + '_' + df_CCFI['Index'])
  df_CCFI.insert(0, 'IsOceanFreight', 'True')
```

Description

1 Setting the URL information where the index data is displayed.

2 Getting information about webpage using Python's built-in module, 'urllib'.

3 Parsing HTML using 'html.parser'

4 Getting the value of the tag information of the desired location

5 Storing the imported value in the text_(index) variable, using 'make2d'

6 Converting the saved data into data frame format

7 Converting the data frame format to a standardized format

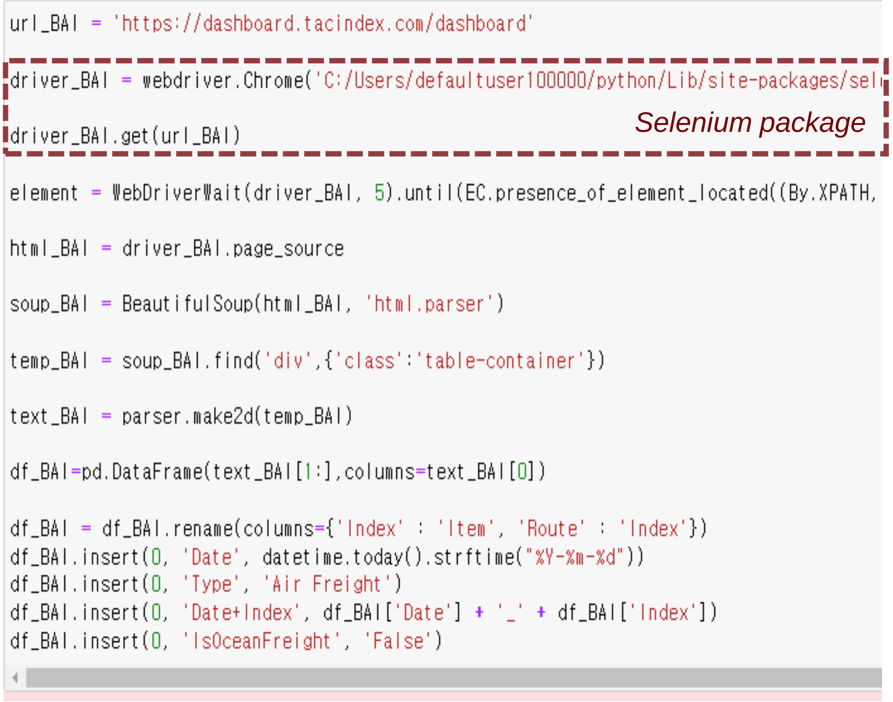
Collecting & Processing

Collecting ▸ Processing ▸ Storing ▸ Querying

- In the case of BAI, BDI, and SCFI, since there is an index value on the dynamic homepage, the Selenium package was additionally used to scrap the data.

Description of 'Collecting & Processing' Process

Code



```
1  url_BAI = 'https://dashboard.tacindex.com/dashboard'

2  driver_BAI = webdriver.Chrome('C:/Users/defaultuser100000/python/Lib/site-packages/sel

3  driver_BAI.get(url_BAI)

4  element = WebDriverWait(driver_BAI, 5).until(EC.presence_of_element_located((By.XPATH,

5  html_BAI = driver_BAI.page_source

6  soup_BAI = BeautifulSoup(html_BAI, 'html.parser')

7  temp_BAI = soup_BAI.find('div',{'class':'table-container'})

8  text_BAI = parser.make2d(temp_BAI)

9  df_BAI=pd.DataFrame(text_BAI[1:],columns=text_BAI[0])

10 df_BAI = df_BAI.rename(columns={'Index' : 'Item', 'Route' : 'Index'})
   df_BAI.insert(0, 'Date', datetime.today().strftime("%Y-%m-%d"))
   df_BAI.insert(0, 'Type', 'Air Freight')
   df_BAI.insert(0, 'Date+Index', df_BAI['Date'] + '_' + df_BAI['Index'])
   df_BAI.insert(0, 'IsOceanFreight', 'False')
```

Selenium package

Description

1 Setting the URL information where the index data is displayed.

2 Specifying the location of the chrome driver, After creating an object with the url_(index) variable.

3 Accessing the URL

4 Waiting up to 5 seconds for web support to load, In the case of 'explicitly wait', it is a command to wait until the part I need is displayed.

5 Getting information about the webpage.

6 Parsing html using 'html.parser'

7 Getting the value of the tag information of the desired location

8 Storing the imported value in the text_(index) variable, using 'make2d'

9 Converting the saved data into a data frame format

10 Converting the data frame format to a standardized format

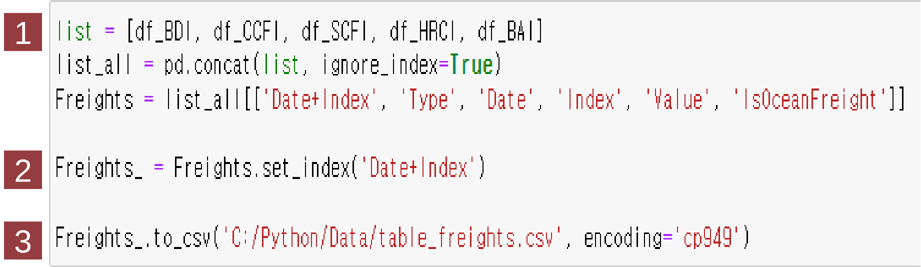
Collecting & Processing

Collecting ▸ Processing ▸ Storing ▸ Querying

- The team merged each index data into one and saved it as a CSV / JSON file.

Description of 'Collecting & Processing' Process

Code

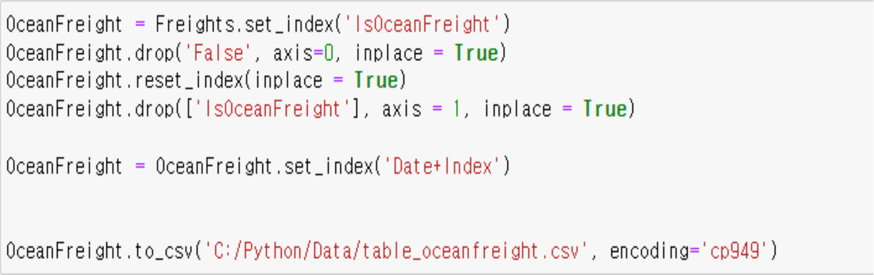


1
```
list = [df_BDI, df_CCFI, df_SCFI, df_HRCI, df_BAI]
list_all = pd.concat(list, ignore_index=True)
Freights = list_all[['Date+Index', 'Type', 'Date', 'Index', 'Value', 'IsOceanFreight']]
```

2
```
Freights_ = Freights.set_index('Date+Index')
```

3
```
Freights_.to_csv('C:/Python/Data/table_freights.csv', encoding='cp949')
```

4

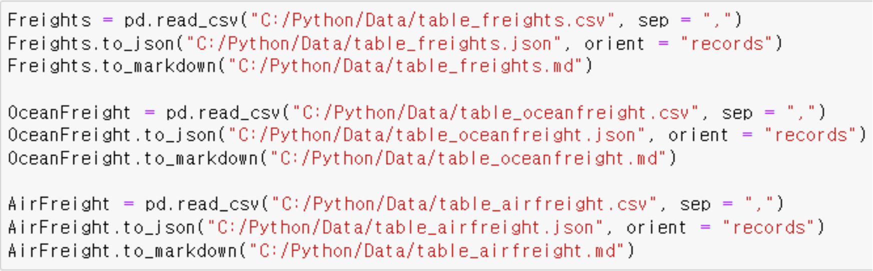


```
OceanFreight = Freights.set_index('IsOceanFreight')
OceanFreight.drop('False', axis=0, inplace = True)
OceanFreight.reset_index(inplace = True)
OceanFreight.drop(['IsOceanFreight'], axis = 1, inplace = True)

OceanFreight = OceanFreight.set_index('Date+Index')


OceanFreight.to_csv('C:/Python/Data/table_oceanfreight.csv', encoding='cp949')
```

5



```
Freights = pd.read_csv("C:/Python/Data/table_freights.csv", sep = ",")
Freights.to_json("C:/Python/Data/table_freights.json", orient = "records")
Freights.to_markdown("C:/Python/Data/table_freights.md")

OceanFreight = pd.read_csv("C:/Python/Data/table_oceanfreight.csv", sep = ",")
OceanFreight.to_json("C:/Python/Data/table_oceanfreight.json", orient = "records")
OceanFreight.to_markdown("C:/Python/Data/table_oceanfreight.md")

AirFreight = pd.read_csv("C:/Python/Data/table_airfreight.csv", sep = ",")
AirFreight.to_json("C:/Python/Data/table_airfreight.json", orient = "records")
AirFreight.to_markdown("C:/Python/Data/table_airfreight.md")
```

Description

1 Merging data frames for each index into one (All Freight indexes)

2 When saving as CSV, Setting the index information so that the index column is not saved.

3 Saving the data frame as a CSV file

4 Proceeding in the same way for the Oceanfreight table and Airfreight table.

5 Saving the data frame as a JSON file

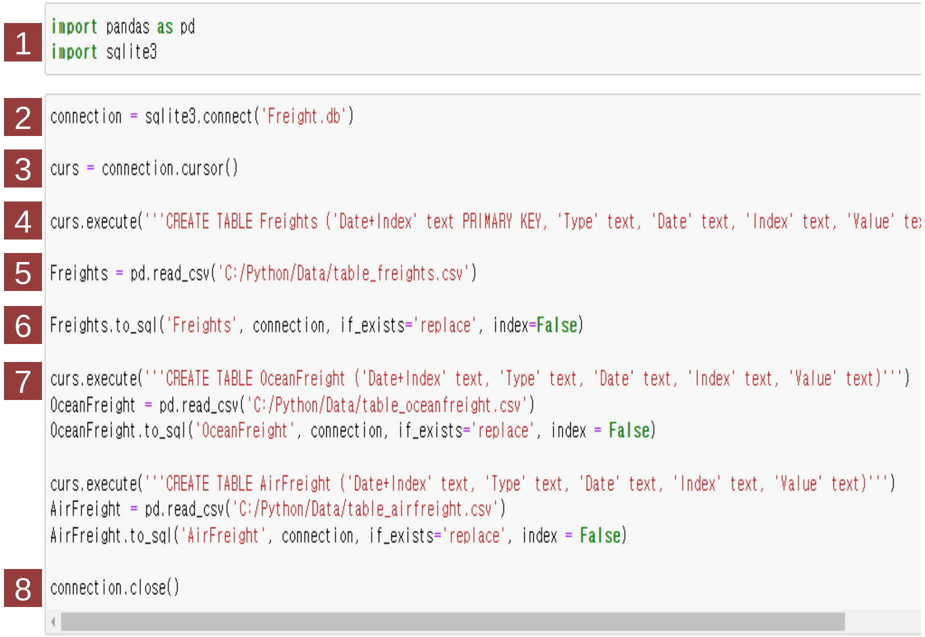
Storing

Collecting ▸ Processing ▸ Storing ▸ Querying

- The team created a table in Python by creating a database of SQLite and then importing the CSV file stored in the previous step.

Description of 'Storing' Process

Code



```
1 import pandas as pd
  import sqlite3

2 connection = sqlite3.connect('Freight.db')

3 curs = connection.cursor()

4 curs.execute('''CREATE TABLE Freights ('Date+Index' text PRIMARY KEY, 'Type' text, 'Date' text, 'Index' text, 'Value' te

5 Freights = pd.read_csv('C:/Python/Data/table_freights.csv')

6 Freights.to_sql('Freights', connection, if_exists='replace', index=False)

7 curs.execute('''CREATE TABLE OceanFreight ('Date+Index' text, 'Type' text, 'Date' text, 'Index' text, 'Value' text)''')
  OceanFreight = pd.read_csv('C:/Python/Data/table_oceanfreight.csv')
  OceanFreight.to_sql('OceanFreight', connection, if_exists='replace', index = False)

  curs.execute('''CREATE TABLE AirFreight ('Date+Index' text, 'Type' text, 'Date' text, 'Index' text, 'Value' text)''')
  AirFreight = pd.read_csv('C:/Python/Data/table_airfreight.csv')
  AirFreight.to_sql('AirFreight', connection, if_exists='replace', index = False)

8 connection.close()
```

Description

1 Importing the necessary packages

■ **Packages**

- Pandas
  - It is a package required to load CSV files into a Panda's data frame.
- Sqlite3
  - It is a package that allows to utilize the functions of SQLite in Python.

2 Connecting/creating to database

3 Creating a cursor object

4 Executing a query that creates a 'Freights' table

5 Loading CSV data into a pandas data frame

6 Writing the data to a SQLite DB table

7 Loading another CSV file into the databases (Oceanfreight, Airfreight indexes)

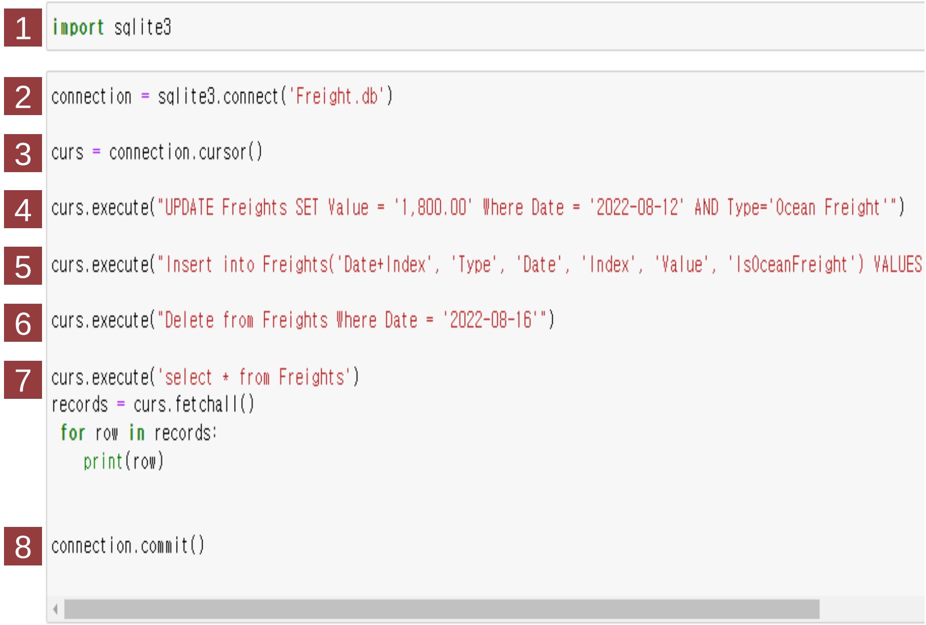
8 Closing connection to SQLite database

Querying

- The team ran queries such as Create, Select, Update, Delete, and Insert in Python.

Description of 'Querying' Process

Code



1
```
import sqlite3
```

2
```
connection = sqlite3.connect('Freight.db')
```

3
```
curs = connection.cursor()
```

4
```
curs.execute("UPDATE Freights SET Value = '1,800.00' Where Date = '2022-08-12' AND Type='Ocean Freight'")
```

5
```
curs.execute("Insert into Freights('Date+Index', 'Type', 'Date', 'Index', 'Value', 'IsOceanFreight') VALUES
```

6
```
curs.execute("Delete from Freights Where Date = '2022-08-16'")
```

7
```
curs.execute('select * from Freights')
records = curs.fetchall()
 for row in records:
    print(row)
```

8
```
connection.commit()
```

Description

1 Importing the necessary packages

■ **Package**
- Sqlite3
  - It is a package that allows to utilize the functions of SQLite in Python.

2 Connecting/creating to database

3 Creating a cursor object

4 Running Update query

5 Running Insert query

6 Running Delete query

7 Running Select query

8 Closing connection to SQLite database

※ In the case of Create query, the table was created in the 'Storing' step

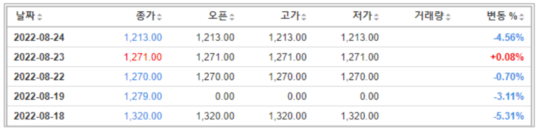
Collecting & Processing

- The team converted the index data from different formats on each web page into a standardized format CSV/JSON file.

Deliverables of Project

Raw data (Freight indexes)

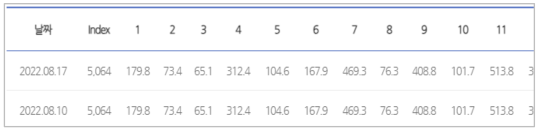
BDI (Baltic Dry Index)



날짜	종가	오픈	고가	저가	거래량	변동 %
2022-08-24	1,213.00	1,213.00	1,213.00	1,213.00		-4.56%
2022-08-23	1,271.00	1,271.00	1,271.00	1,271.00		+0.08%
2022-08-22	1,270.00	1,270.00	1,270.00	1,270.00		-0.70%
2022-08-19	1,279.00	0.00	0.00	0.00		-3.11%
2022-08-18	1,320.00	1,320.00	1,320.00	1,320.00		-5.31%

(https://kr.investing.com/indices/baltic-dry-historical-data)

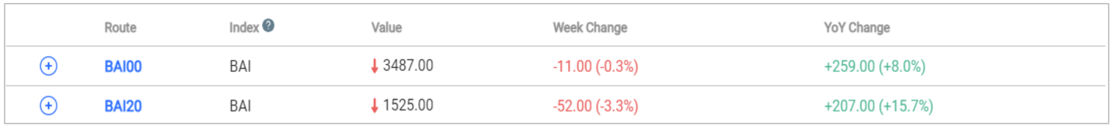
HRCI (Howe Robinson Container Index)



날짜	Index	1	2	3	4	5	6	7	8	9	10	11	
2022.08.17	5,064	179.8	73.4	65.1	312.4	104.6	167.9	469.3	76.3	408.8	101.7	513.8	3
2022.08.10	5,064	179.8	73.4	65.1	312.4	104.6	167.9	469.3	76.3	408.8	101.7	513.8	3

(https://www.ksg.co.kr/shippingGraph/hrci_graph.jsp)

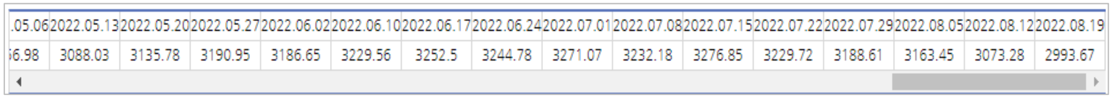
BAI (Baltic Air Freight Indices)



	Route	Index	Value	Week Change	YoY Change
⊕	BAI00	BAI	↓ 3487.00	-11.00 (-0.3%)	+259.00 (+8.0%)
⊕	BAI20	BAI	↓ 1525.00	-52.00 (-3.3%)	+207.00 (+15.7%)

(https://dashboard.tacindex.com/dashboard)

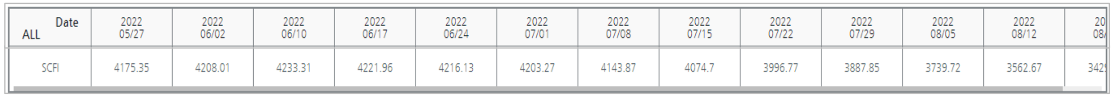
CCFI (China Containerized Freight Index)



.05.06	2022.05.13	2022.05.20	2022.05.27	2022.06.02	2022.06.10	2022.06.17	2022.06.24	2022.07.01	2022.07.08	2022.07.15	2022.07.22	2022.07.29	2022.08.05	2022.08.12	2022.08.19
6.98	3088.03	3135.78	3190.95	3186.65	3229.56	3252.5	3244.78	3271.07	3232.18	3276.85	3229.72	3188.61	3163.45	3073.28	2993.67

(https://www.kcla.kr/web/inc/html/4-1_2.asp)

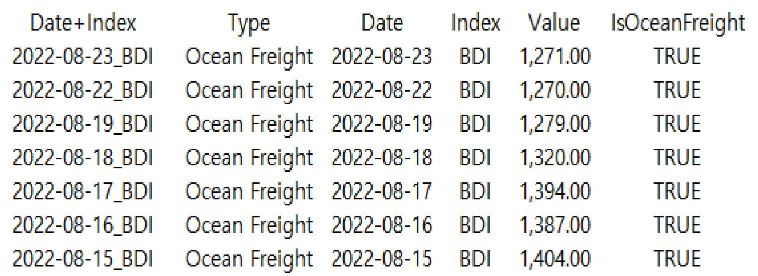
SCFI (Shanghai Containerized Freight Index)



Date ALL	2022 05/27	2022 06/02	2022 06/10	2022 06/17	2022 06/24	2022 07/01	2022 07/08	2022 07/15	2022 07/22	2022 07/29	2022 08/05	2022 08/12	20 08/
SCFI	4175.35	4208.01	4233.31	4221.96	4216.13	4203.27	4143.87	4074.7	3996.77	3887.85	3739.72	3562.67	342

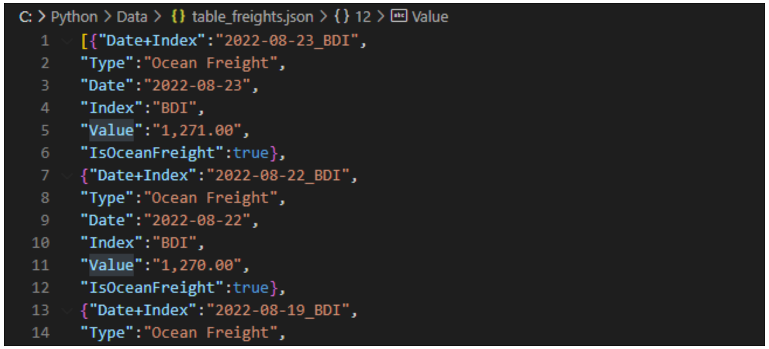
(https://www.tradlinx.com/freight-index)

CSV / JSON file



Date+Index	Type	Date	Index	Value	IsOceanFreight
2022-08-23_BDI	Ocean Freight	2022-08-23	BDI	1,271.00	TRUE
2022-08-22_BDI	Ocean Freight	2022-08-22	BDI	1,270.00	TRUE
2022-08-19_BDI	Ocean Freight	2022-08-19	BDI	1,279.00	TRUE
2022-08-18_BDI	Ocean Freight	2022-08-18	BDI	1,320.00	TRUE
2022-08-17_BDI	Ocean Freight	2022-08-17	BDI	1,394.00	TRUE
2022-08-16_BDI	Ocean Freight	2022-08-16	BDI	1,387.00	TRUE
2022-08-15_BDI	Ocean Freight	2022-08-15	BDI	1,404.00	TRUE

⋮



```
[{"Date+Index":"2022-08-23_BDI",
"Type":"Ocean Freight",
"Date":"2022-08-23",
"Index":"BDI",
"Value":"1,271.00",
"IsOceanFreight":true},
{"Date+Index":"2022-08-22_BDI",
"Type":"Ocean Freight",
"Date":"2022-08-22",
"Index":"BDI",
"Value":"1,270.00",
"IsOceanFreight":true},
{"Date+Index":"2022-08-19_BDI",
"Type":"Ocean Freight",
```

⋮

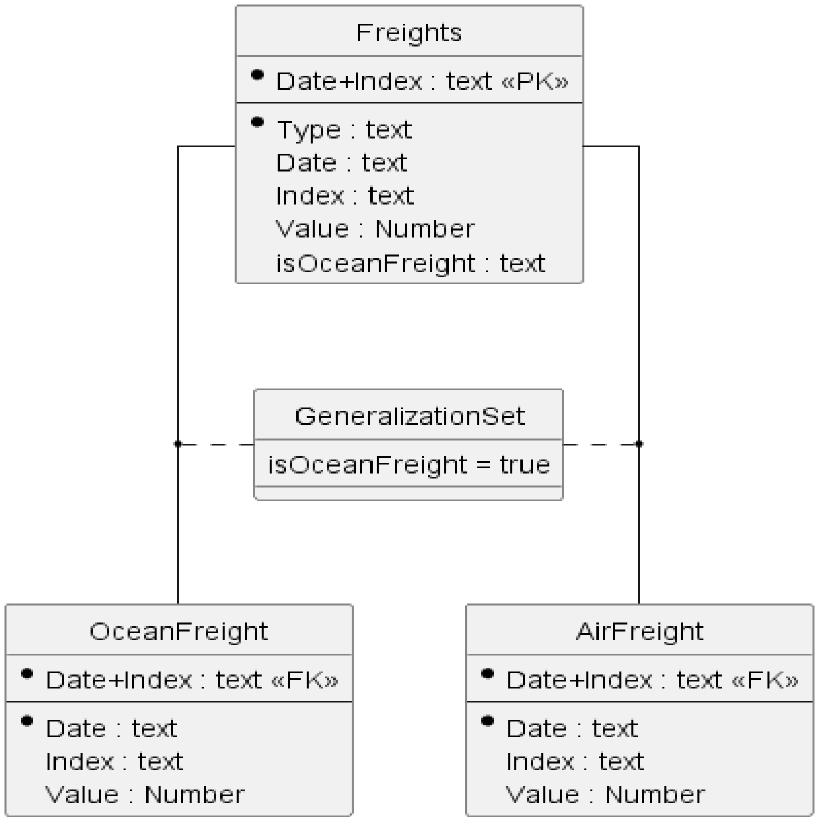
- CSV / JSON file for a total of 5 index information including BDI, HRCI, CCFI, SCFI, and BAI

Storing & Querying

- The team designed the database in SQLite according to the ERD. As a result, It is possible to search and manage stored in the database by running a query.

Deliverables of Project

Entity Relationship Diagram (ERD)



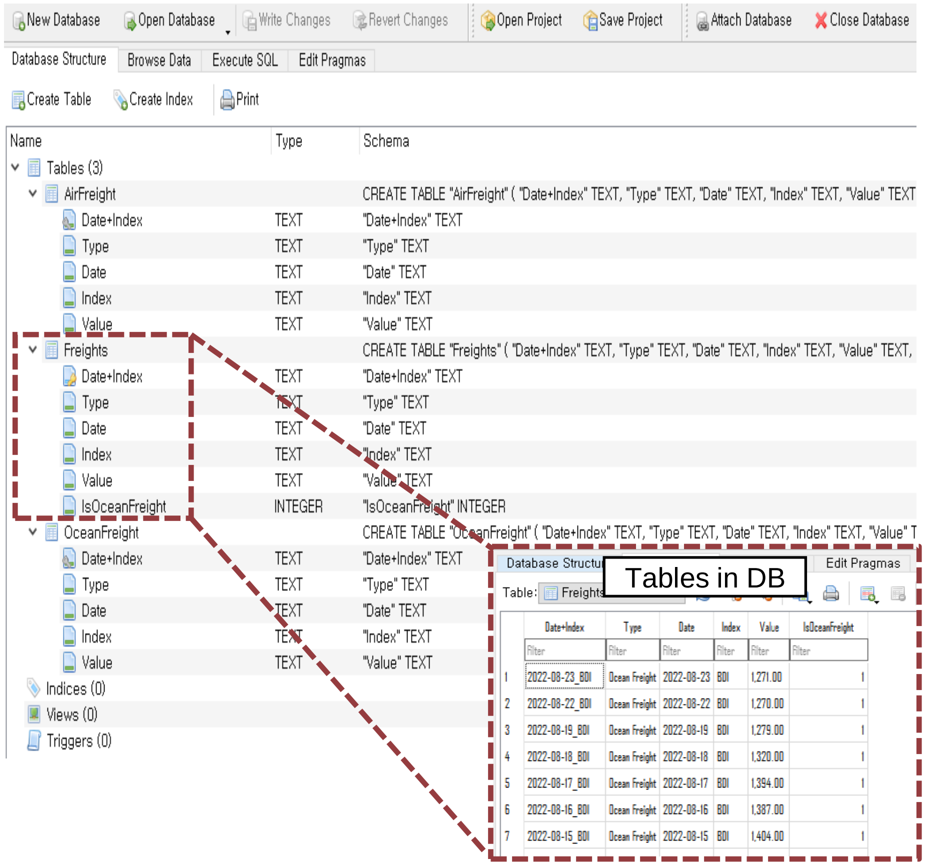
Freights
- Date+Index : text «PK»
- Type : text
  Date : text
  Index : text
  Value : Number
  isOceanFreight : text

GeneralizationSet
isOceanFreight = true

OceanFreight
- Date+Index : text «FK»
- Date : text
  Index : text
  Value : Number

AirFreight
- Date+Index : text «FK»
- Date : text
  Index : text
  Value : Number

Database in SQLite



Name	Type	Schema
Tables (3)		
AirFreight		CREATE TABLE "AirFreight" ("Date+Index" TEXT, "Type" TEXT, "Date" TEXT, "Index" TEXT, "Value" TEXT
Date+Index	TEXT	"Date+Index" TEXT
Type	TEXT	"Type" TEXT
Date	TEXT	"Date" TEXT
Index	TEXT	"Index" TEXT
Value	TEXT	"Value" TEXT
Freights		CREATE TABLE "Freights" ("Date+Index" TEXT, "Type" TEXT, "Date" TEXT, "Index" TEXT, "Value" TEXT,
Date+Index	TEXT	"Date+Index" TEXT
Type	TEXT	"Type" TEXT
Date	TEXT	"Date" TEXT
Index	TEXT	"Index" TEXT
Value	TEXT	"Value" TEXT
IsOceanFreight	INTEGER	"IsOceanFreight" INTEGER
OceanFreight		CREATE TABLE "OceanFreight" ("Date+Index" TEXT, "Type" TEXT, "Date" TEXT, "Index" TEXT, "Value" T
Date+Index	TEXT	"Date+Index" TEXT
Type	TEXT	"Type" TEXT
Date	TEXT	"Date" TEXT
Index	TEXT	"Index" TEXT
Value	TEXT	"Value" TEXT
Indices (0)		
Views (0)		
Triggers (0)		

Tables in DB

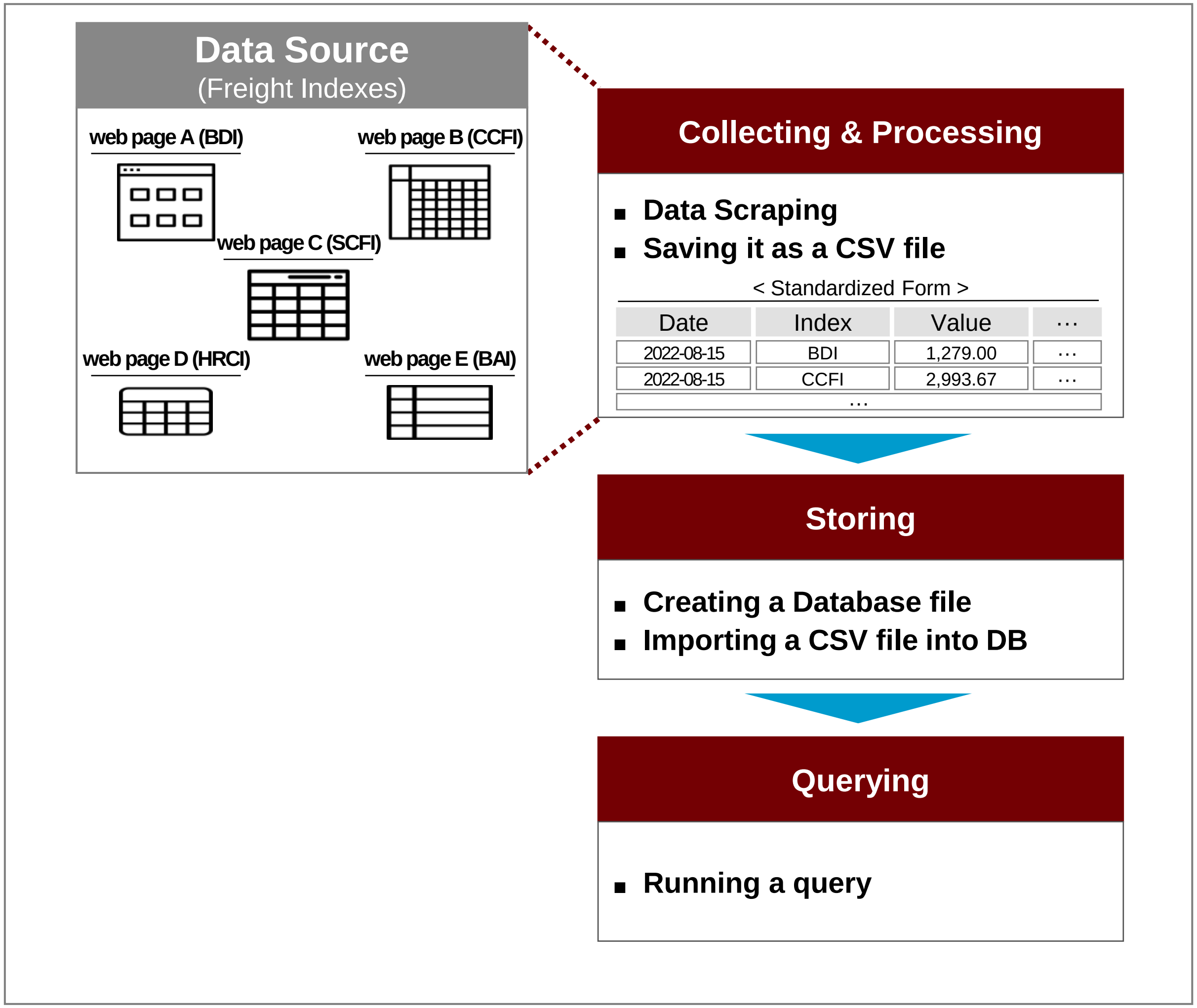
Table: Freights

	Date+Index	Type	Date	Index	Value	IsOceanFreight
	Filter	Filter	Filter	Filter	Filter	Filter
1	2022-08-23_BDI	Ocean Freight	2022-08-23	BDI	1,271.00	1
2	2022-08-22_BDI	Ocean Freight	2022-08-22	BDI	1,270.00	1
3	2022-08-19_BDI	Ocean Freight	2022-08-19	BDI	1,279.00	1
4	2022-08-18_BDI	Ocean Freight	2022-08-18	BDI	1,320.00	1
5	2022-08-17_BDI	Ocean Freight	2022-08-17	BDI	1,394.00	1
6	2022-08-16_BDI	Ocean Freight	2022-08-16	BDI	1,387.00	1
7	2022-08-15_BDI	Ocean Freight	2022-08-15	BDI	1,404.00	1

Project Benefits

- The team built a data processing pipeline for freight indexes, which is expected to ensure the work efficiency in the pricing process and data accumulation

Data Pipeline



Data Source
(Freight Indexes)

Collecting & Processing

- **Data Scraping**
- **Saving it as a CSV file**

< Standardized Form >

Date	Index	Value	…
2022-08-15	BDI	1,279.00	…
2022-08-15	CCFI	2,993.67	…
…			

Storing

- **Creating a Database file**
- **Importing a CSV file into DB**

Querying

- **Running a query**

Benefits

Work efficiency in the pricing process

- The efficiency of pricing tasks will be increased as index data can be viewed and managed in one integrated channel.

Data Accumulation

- By storing an index for each web page in the database, it becomes possible to develop models and strategies using historical data in the future.