# Introduction, Python and SQL
Note 1

Financial Database
BAF 507E

Inmoo Lee

KAIST

Fall (1st half) 2025

## Table of Contents

## Introduction

- Course Overview
- Introduction to Python

## Financial Database

- Required Materials
  - Various materials available on the course web page
- Grades
  - Exam (50%), Bloomberg Certificate (20%), Final Group Project and Presentation (20%) and Participation and Attendance (10%)
- Exam
  - In-class exam

## Financial Databases to be Covered

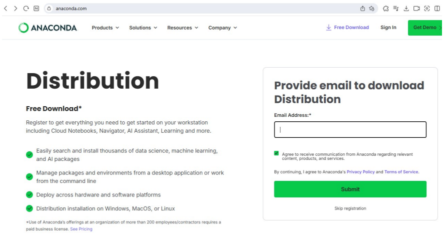
- LSEG Workspace
  - LSEG Workspace is accessible from the terminals at the Reuters Trading Center and the KOSCOM Data Center on the 3rd floor.
  - In addition, you can access to it anywhere with available ID and PASSWORD (web-based).
  - You can access to LSEG Workspace through internet by signing in here (web access) or by downloading and installing the program and signing in (`https://www.lseg.com/en/data-analytics/products/workspace/download-workspace`).
- Bloomberg
  - Similar to LSEG Workspace, it provides a global real-time and historical financial & economic data as well news
  - Bloomberg terminals are available at the KOSCOM Data Center on the 3rd floor.

## Bloomberg Market Concepts

- As explained in the syllabus, you are required to finish all BMC (Bloomberg Market Concepts) courses (Economic Indicators, Currencies, Fixed Income and Equities) and submit the certificate by **September 30**. Terminal Basics is not required for the certificate, but I strongly encourage all of you to finish it for better use of Bloomberg.
- You should sign up Bloomberg for Education with your own name and email address so that the certificate will be **under your name**
  `https://portal.bloombergforeducation.com/sign_up`.
- When you login after your sign up, please use the class code, **"QBJLZWNRDX"**, so that I can monitor your progresses.
- Please start it **as soon as possible**!!!
- This is an **individual** work.

## Python

- Python is an open-source programming language (`https://www.python.org/about/`).
- You can install and use Python through Anaconda(`https://www.anaconda.com/download`).
- All Python codes for the classes are in the Jupyter Notebook format (ipynb). You can run these in Jupyter/JupyterLab/MS Visual Studio Code.



Distribution

Free Download*

Register to get everything you need to get started on your workstation including Cloud Notebooks, Navigator, AI Assistant, Learning and more.

- Easily search and install thousands of data science, machine learning, and AI packages
- Manage packages and environments from a desktop application or work from the command line
- Deploy across hardware and software platforms
- Distribution installation on Windows, MacOS, or Linux

*Use of Anaconda's offerings at an organization of more than 200 employees/contractors requires a paid business license. See Pricing

Provide email to download Distribution

Email Address:*

[✓] Agree to receive communication from Anaconda regarding relevant content, products, and services.

By continuing, I agree to Anaconda's Privacy Policy and Terms of Service.

Submit

Skip registration

## JupyterLab Basics

- Code and Markdown
  - Each cell can contain codes or texts.
  - The default is code but you can change it to Markdown to include comments in a separate cell. ([ESC + M], click here for more information on Markdown)
- How to run?
  - Typically, you can select a cell or cells and run selected cells ([Control + Enter])
  - You can also run line by line ([F9]). However, in this case, you are going to use a separate console (click here for more information on console)
- Add or split cells
  - You can add a new cell below ([ESC + B]) or above ([ESC + A]) the current cell.
  - You can split cell at a location where the cursor is ([CTRL + Shift + -]).

## Microsoft Visual Studio Code

- You can download it from `https://code.visualstudio.com/` after installing Python through Anaconda or from Python.org
- Before using VS Code, you need to install Python extension within VS Code. Check this site for more details.
- You can use Jupyter Notebook. Check this site for more information. There are VS Code Jupyter extension for Jupyter notebook support.
- During the class, I will use VS Code for demonstration but you can use others that you are more familiar with for your own exerclse.

## MS Visual Studio Code Basics

- Code and Markdown
  - Each cell can contain codes or texts.
  - +Code or +Markdown tabs can be used to make a new one.
- How to run?
  - You can use ▷ tab to run each cell or use the Run all tab to run all cells
  - You can also use ([Control + Enter]) to run a cell.
  - You can also run line by line ([F10] or tab).
- Add or split cells
  - In addition to using tabs to add or split cells, you can also use keys to add a new cell below ([ESC + B]) or above ([ESC + A]) the current cell.
  - Likewise, you can split cell at a location where the cursor is using keys ([CTRL + Shift + -]) or using menu bars to split or join cells.

## Python: General Information

- To create a new variable, vector, matrix or data, use " = "
- To add comments, use "#" inside a Code cell
- You need to install "packages" before using those packages (click here to find how you can install packages). You can install within JupyterLab by running '!pip install PACKAGE_NAME'.
- Once you install a package, use "import ***", where *** is the name of the package.
- Many resources are available online: (`https://wiki.python.org/moin/BeginnersGuide/Programmers`)
- Check the following for the basics on Python, Numpy, Scipy and Matplotlib
  - (`https://cs231n.github.io/python-numpy-tutorial/`)

## Getting Data into Python

- Ways to get the data into Python
  - Input within a program
  - Import from an external data source
- Check FDNote1W2025.ipynb

## Working directory

- You can check the current working directory by using
  *import os*
  *os.getcwd()*
- To change the working directory, you can use
  *os.chdir(' ')*
- To permanently save the data, you can use
  *NAME1.to_feather('NAME2.ft')*
  where NAME1 is the name of a dataframe created in Python and NAME2 is the name of the file to be stored in the current working directory.

## Alternative file formats

- The following compares different file formats to be used to store dataframes in Python.

| Feature | CSV | Excel | JSON | HDF5 | Feather | Parquet | Pickle |
|---|---|---|---|---|---|---|---|
| Human-Readable | ✓ | ✓ | ✓ | ✗ | ✗ | ✗ | ✗ |
| Interoperability | ✓ (high) | ✓ (med) | ✓ (high) | ✓ (low) | ✓ (high) | ✓ (high) | ✗ |
| Speed (I/O) | slowest | slow | slow | fast | fastest | fast | fast |
| File Size | largest | large | large | small | small | smallest | medium |
| Data Types Preserved | ✗ (infer) | ✗ (infer) | ✗ (infer) | ✓ | ✓ | ✓ | ✓ |
| Columnar Storage | ✗ | ✗ | ✗ | ✗ | ✓ | ✓ | ✗ |
| Big Data Friendly | ✗ | ✗ | ✗ | ✓ | ✓ | ✓ | ✗ |
| Security Risk | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✓ (high) |

Table: Generated by Gemini: Comparison of Different Data Storage Formats

## SQL

- Stands for Structured Query Language
- GUI (Graphical User Interface) interfaces are often available.
- Interfaces to many programming languages: R, python, perl, PHP, etc.
- There are a few alternatives (e.g., sqlite3 and pandasql packages) to run SQL in Python. In this course, we will focus on 'sqldf' in the *pandasql* package that directly uses Pandas dataframes[1]

[1] *sqlite3* works with sql tables and therefore, it requires one to convert a pandas dataframe to a sql table within a database before using it, whereas *sqldf* directly works with dataframes.

## Databases vs. Tables

- A database server can contain many databases
- Databases are collections of data tables
- Tables are two-dimensional with rows (observations) and columns (variables)
- Limited mathematical and summary operations available
- SQL is very convenient to use in combining information from multiple tables

## Select

- In many cases, all you need to do with databases is to select some subsets of variables and/or observations from a table (or across tables), and use some other programs (such as SAS or Python) to manipulate them. In SQL, the **SELECT** statement is the workhorse for these operations.
  - SELECT columns or computations
    FROM table
    WHERE condition
    GROUP BY columns
    ORDER BY column

## Summaries and Computations

- SQL supports basic arithmetic operations to create new columns, as well as some summarization functions that include
  - COUNT()
  - AVG() (mean)
  - SUM()
  - MIN()
  - MAX()
  - STD()
  - STDERR()
- In sqldf, STD and STDERR do not work

## pandasql

- If you use sqldf in *pandasql*, you can directly work with pandas dataframes even though it is powered by SQLite3

```
from pandasql import sqldf

query='''select a.**, a.***, ...
            from logret as a
            ...
            '''
df=sqldf(query,locals())
```

- You can use either locals() or globals() depending on the scope of names (variables) defined (whether they are visible throughout the module or only inside a function). You can use the pysqldf() function defined in the note for a simpler use of sqldf.)

## Use of a function

- Oftentimes, it is convenient to define and use a customized function.
- For example, rather than using sqldf(query, locals()) to run sqldf, you can define a new function that has only one argument as below

```
def pysqldf(q):
    return sqldf(q, globals())[2]

pysqldf(query)
```

[2] If you use locals() instead of globals(), you will get error message when you run this function. This is because the table (dataframe) used inside a query statement will not be recognized since it is defined outside the function. Therefore, you have to use globals() when you define a function.

## SQL: How to summarize values of columns?

- Find out the average of stock returns for each stock using SQL.

## Exercise: To be covered next week

- Import "note2data.xlsx" into Python
    - The data include monthly stock returns of three companies, Microsoft, IBM and Walmart as well as S&P 500 index returns and others
- Try to do the following using Python
    - Calculate mean, minimum, maximum and standard deviation of returns of three stocks.
    - Define a category variable, *isign*, to indicate positive and negative returns.
    - Check the frequency of positive and negative returns for each stock.
    - Create another categorical variable to indicate each calendar year.
    - Generate frequency tables of stock returns for both categories.