금융 빅데이터 데이터 특징 및 자료 수집

목차

금융 빅데이터 수집 및 분석 개발 환경

■ 환경:

- Python(>=3.10)
- VS Code 또는 Jupyter (Google Colab 이용 가능)
- .env 사용

금융 빅데이터

1. 데이터의 양 (Volume)

- 방대한 규모
  - 주식 거래, 신용카드 결제, 은행 거래 등 모든 금융 활동이 데이터로 기록됩니다. 전 세계적으로 매일 수십억 건의 금융 데이터가 생성됩니다.

- 지속적 증가
  - 핀테크, 모바일 뱅킹, 비대면 거래가 확산되면서 데이터 생성 속도가 기하급수적으로 늘어나고 있습니다.

금융 데이터의 본질: 5V + α

2. 5V

(1) 데이터의 양 (Volume)

- 방대한 규모: 주식 거래, 신용카드 결제, 은행 거래 등 모든 금융 활동이 데이터로 기록됩니다. 전 세계적으로 매일 수십억 건의 금융 데이터가 생성됩니다.
- 지속적 증가: 핀테크, 모바일 뱅킹, 비대면 거래가 확산되면서 데이터 생성 속도가 기하급수적으로 늘어나고 있습니다.

(2) 데이터의 속도 (Velocity)

- 실시간성: 주식 시장, 외환 시장 등은 1초에 수백만 건의 데이터가 발생하며, 실시간으로 분석하고 의사결정을 내려야 합니다.
- 높은 빈도: 고빈도 매매(High-Frequency Trading)와 같이 초 단위의 정밀한 데이터 처리가 중요합니다.

금융 데이터의 본질: 5V + α

2. 5V

(3) 데이터의 다양성 (Variety)

- 정형 데이터: 주가, 거래량, 계좌 잔액, 신용 등급 등 데이터베이스 형태로 깔끔하게 정리된 데이터입니다.
- 비정형/반정형 데이터: 뉴스 기사, SNS 게시물, 공시 보고서(PDF, HWP 등), 고객 상담 녹취록, 이메일 등 형태가 정해져 있지 않거나 부분적으로 구조화된 데이터입니다.

(4) 데이터의 정확성 (Veracity)

- 높은 신뢰도: 금융 데이터는 정확한 거래 기록을 기반으로 하므로 일반적으로 신뢰도가 매우 높습니다.

* 오류의 치명성: 단 하나의 데이터 오류도 금융 거래에 심각한 손실을 초래할 수 있으므로, 데이터의 무결성과 정확성이 무엇보다 중요합니다.

금융 데이터의 본질: 5V + α

2. 5V

(5) 데이터의 가치 (Value)

- 높은 가치: 금융 빅데이터는 시장 예측, 신용 위험 평가, 사기 탐지, 고객 맞춤형 서비스 제공 등 막대한 경제적 가치를 창출합니다.
- 정확한 분석의 중요성: 데이터의 양이 많을수록 잘못된 분석은 큰 손해로 이어질 수 있어, 양질의 데이터 분석 역량이 필수적입니다.

(6) 데이터의 보안 (Security)

- 최고 수준의 보안 요구: 개인의 금융 정보와 기업의 민감한 자산 정보가 포함되어 있어, 데이터 유출 시 막대한 피해가 발생할 수 있습니다.
- 엄격한 규제: 각국의 금융 규제 기관(예: 한국 금융감독원)은 금융 데이터의 저장, 전송, 활용에 대해 매우 엄격한 보안 및 개인정보보호 규정을 적용합니다.

수집 전략: ‘API 우선’ 원칙

- **API**
  - 스키마/문서/버전 정책/쿼터 등 체계적 → 안정적/확장 가능한 파이프라인.

- **크롤링**
  - 구조변경·차단·법/약관 이슈, 운영 리스크 증가 → 예외적 최소화 권장.

주요 데이터 소스 개관

- **공공:**
  - 한국은행 ECOS(거시지표), 금융감독원 DART(전자공시/재무/지배구조 등).
- **민간/해외:**
  - Alpha Vantage(시세/지표),
  - …
- **커뮤니티 라이브러리:**
  - yfinance(야후 파이낸스 기반, 키 불필요).
- **국내 포털:**
  - 네이버 금융(공식 Open API 부재 → 약관/구조변경 리스크).

한국은행 ECOS Open API: 키 발급 · 등록

1) 접속: 한국은행 ECOS(Open API) 홈페이지 → 회원가입.

2) 'Open API' 메뉴에서 인증키 발급 신청(개인/기관 선택).

3) 발급 후 마이페이지에서 인증키 확인, 일일 쿼터·초당 제한 숙지.

4) 요청 규격: REST(JSON/XML), 기간(YYYYMM/분기/연간)·통계표 코드·항목 코드.

5) 실무 권장: .env에 ECOS_API_KEY 저장, 코드에서는 os.getenv로 주입.

한국은행 Open API 서비스 https://ecos.bok.or.kr/api/#/DevGuide/StatisticalCodeSearch

ECOS 실무 호출 · 메타데이터 포인트

- **통계표/항목 코드가 핵심: 동일 지표라도 항목(지역/유형/계정)으로 값이 달라질 수 있음.**
- **요청 파라미터 예: 인증키, 통계표코드, 주기, 시작/끝 기간, 항목코드 등.**
- **메타 관리: 단위(지수/%)·주기(월/분기/연간)·계절조정 여부, 개편 이력 기록**
- **오류/쿼터: 400(파라미터), 401(인증), 429(쿼터 초과) → 백오프·캐시·증분 수집.**

[기본 구조]
http(s)://ecos.bok.or.kr/api/StatisticSearch

- **http(s)** : http:// 또는 https:// 프로토콜.
  - 일부 환경(특히 Colab)에서 SSL 핸드셰이크 문제가 있을 때 http://를 권장.
- **ecos.bok.or.kr** : 한국은행 ECOS(Open API) 서버 도메인
- **/api/StatisticSearch** : 통계 조회**(StatisticSearch)** API 엔드포인트

ECOS 실무 호출 · 메타데이터 포인트

Path 파라미터별 의미

위 치	예 시	설 명
{api_key}	인증키	한국은행 오픈API 발급 받은 인증키 (개인마다 다름).
json	json	응답 데이터 포맷 (json/xml 중 선택 가능, 보통 json).
kr	kr	언어 코드 (한국어). en으로 요청하면 영어 응답.
1	1	시작 행 번호 (페이징). 1부터 시작.
9999	9999	끝 행 번호 (페이징). 한 번에 가져올 최대 레코드 수. (최대 9999)
{stat_code}	902Y007	통계코드 : 어떤 통계를 조회할지 지정. (예: 국제 주요국 생산자물가지수)
M	M	주기(Periodicity) : M = 월별, Q = 분기별, A = 연도별, D = 일별 등
{start}	201001	조회 시작 시점 : 주기가 M(월별)이면 YYYYMM 형식
{end}	202206	조회 종료 시점 : 동일 형식
{item_code1}	KR	세부 항목 코드(계정항목) : 예) 국가코드, 품목 등 세부값 지정

ECOS 실무 호출 · 메타데이터 포인트

[통화량(광의통화 M2) 자료]

http://ecos.bok.or.kr/api/StatisticSearch/본인인증키**/json/kr/1/9999/101Y002/M/201801/202312/AA**

위치	값	설명
본인인증키	–	ECOS에서 발급받은 본인 API Key
json	json	응답 포맷 (json)
kr	kr	언어(한국어)
1	1	시작행 번호 (페이징)
9999	9999	끝행 번호 (최대 9999행)
101Y002	101Y002	통화량(광의통화 M2) 통계코드
M	M	주기: 월별 데이터
201801	201801	시작 기간: 2018년 1월
202312	202312	종료 기간: 2023년 12월
AA	AA	계정항목 코드: 전체(Seasonally Unadjusted 총통화) 예시

DART Open API: 키 발급 · 등록

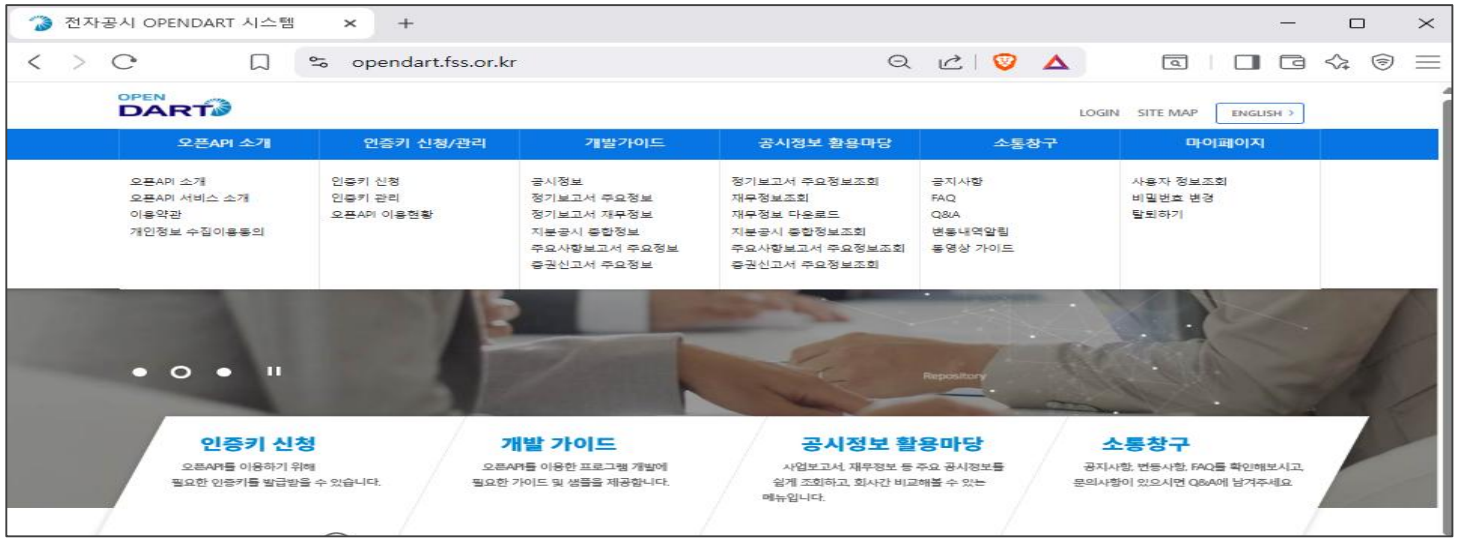
1) 접속: DART(Open API) 포털 → 인증키 신청.

2) 승인 후 API Key 확인(마이페이지). 일/초당 호출 제한 숙지.

3) 주요 엔드포인트: 고유번호 조회, 공시검색, 재무정보, 지배구조, XBRL 등.

4) 실무 권장: .env에 DART_API_KEY 저장, 요청·응답 예시와 에러코드 표를 팀 위키에 정리.

5) 유의: 상장폐지/분할/합병 등 기업 이벤트에 따른 식별자/히스토리 동기화.



https://opendart.fss.or.kr/

DART Open API: 키 발급 · 등록

- **corpCode (기업 고유코드 목록)**
  - 목적: 회사명 → corp_code 식별 (다른 API의 필수 키로 사용)
  - Method / URL
    - GET https://engopendart.fss.or.kr/engapi/corpCode.xml (응답: ZIP 내부 XML)
  - 요청 파라미터
    - crtfc_key (Y): 발급받은 인증키(40자리)
  - 응답(요약): ZIP 안 XML에 corp_code, corp_name, stock_code, modify_date 컬럼 포함
  - 비고: 브라우저에서 ZIP이 .xml로 저장될 수 있으니 확장자를 .zip으로 바꿔 열면 정상

```
GET https://engopendart.fss.or.kr/engapi/corpCode.xml?crtfc_key=YOUR_KEY
```

DART Open API: 키 발급 · 등록

- **list (공시 목록 조회)**

  - 목적: 기간·기업·공시유형으로 전자공시 리스트 조회 → rcept_no 확보

  - Method / URL
    - GET https://engopendart.fss.or.kr/engapi/list.json (또는 .xml)

  - 요청 파라미터 (주요)
    - crtfc_key (Y): 인증키
    - corp_code (N): 기업 고유코드(8자리)
    - bgn_de (N): 시작일(YYYYMMDD) – corp_code 없이 조회 시 기간 3개월 제한
    - end_de (N): 종료일(YYYYMMDD, 기본값=검색일)
    - last_reprt_at (N): 최종보고서만(Y/N), 기본 N
    - pblntf_ty (N): 공시구분(예: A 정기, B 주요사항 등)
    - pblntf_detail_ty (N): 상세구분(예: A001 사업보고서, A002 반기, A003 분기 … 전체 표 제공)
    - corp_cls (N): 법인구분 Y(코스피)/K(코스닥)/N(코넥스)/E(기타)
    - sort / sort_mth (N): 정렬 필드(date/crp/rpt) & 방식(asc/desc, 기본 desc)
    - page_no, page_count (N): 페이지/페이지당건수(1~100, 기본 10)

  - 응답 주요 필드
    - rcept_no(접수번호, 14자리), rcept_dt(접수일), report_nm(보고서명), corp_name, stock_code 등
    - 뷰어 예시 링크: https://englishdart.fss.or.kr/dsbh001/main.do?rcpNo={rcept_no}

DART Open API: 키 발급 · 등록

공시 목록 (삼성전자, 2024-01-01~2025-06-30)

```
GET https://engopendart.fss.or.kr/engapi/list.json
  ?crtfc_key=YOUR_KEY
  &corp_code=########
  &bgn_de=20240101
  &end_de=20250630
  &last_reprt_at=Y
  &pblntf_ty=A
  &page_count=100
```

DART Open API: 키 발급 · 등록

- **fnlttSinglAcntAll (단일회사 전체 재무제표)**

  • 목적: 연/분기 재무제표(BS/IS/CIS/CF/SCE) 전체 계정 라인 수집

  • Method / URL
    - GET https://engopendart.fss.or.kr/engapi/fnlttSinglAcntAll.json (또는 .xml)

  • 요청 파라미터 (필수)
    - crtfc_key (Y): 인증키
    - corp_code (Y): 기업 고유코드(8자리)
    - bsns_year (Y): 사업연도(4자리, 2015년 이후 제공)
    - reprt_code (Y): 보고서 코드
    - 11013 1분기, 11012 반기, 11014 3분기, 11011 사업(연간)
    - fs_div (Y): 연결/별도 구분 — CFS(연결), OFS(별도)

  • 응답 주요 필드
    - sj_div(재무제표 구분: BS/IS/CIS/CF/SCE), account_nm(계정명), thstrm_amount·frmtrm_amount 등, currency, rcept_no 등

DART Open API: 키 발급 · 등록

단일회사 전체 재무제표(연결·연간)

```
GET https://engopendart.fss.or.kr/engapi/fnlttSinglAcntAll.json
  ?crtfc_key=YOUR_KEY
  &corp_code=########
  &bsns_year=2024
  &reprt_code=11011
  &fs_div=CFS
```

Alpha Vantage: 키 발급 · 사용

1) 접속: Alpha Vantage 홈페이지 → 무료 API 키 발급(이메일 인증).

2) 카테고리: 주가(일/분), 외환, 암호화폐, 기술적 지표, 펀더멘털 등.

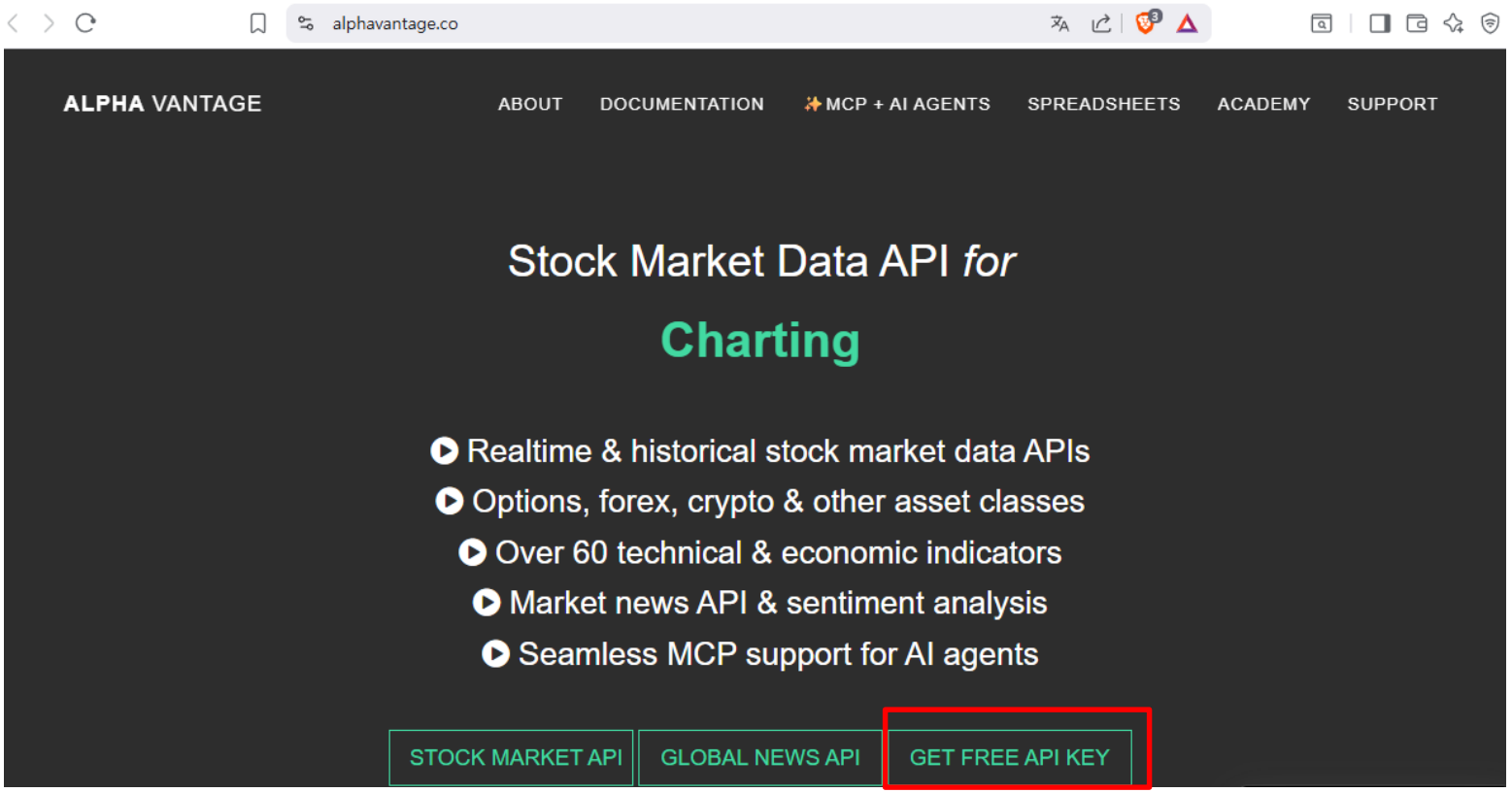
3) 호출 제한: 무료 플랜 초당/분당 쿼터 존재 → 캐시·백오프 권장.

4) 파이썬 예: alpha_vantage 패키지 또는 requests로 직접 호출(함수 래퍼 작성 추천).

5) 대체/보완: 정책 변경 위험 대비, 동일 지표를 2개 이상 소스로 교차검증.

6) 주식, 암호화폐, 외환(FX) 등의 금융 시장 데이터를 실시간 또는 과거 기록으로 접근하여 금융 분석, 알고리즘 트레이딩, 기술적 지표 개발 등 정량적 금융(Quantitative Finance) 분야의 개발에 활용

Alpha Vantage: 키 발급 · 사용



Email 인증 후

Welcome to Alpha Vantage! Here is your API key: 발급 키 Please record this API key at a safe place for future data access.

Alpha Vantage: 키 발급 · 사용

(A) 주식 시계열(Time Series)

- **일/주/월/일봉조정**
  - **Daily:** GET https://www.alphavantage.co/query?function=TIME_SERIES_DAILY
    &symbol=IBM
    &outputsize=full
    &datatype=json
    &apikey=YOUR_KEY

파라미터: function=TIME_SERIES_DAILY, symbol, outputsize=compact|full, datatype=json|csv, apikey

  - **Daily Adjusted:** function=TIME_SERIES_DAILY_ADJUSTED (배당/분할 반영 종가 포함)
  - **Weekly/Monthly:** function=TIME_SERIES_WEEKLY(_ADJUSTED), TIME_SERIES_MONTHLY(_ADJUSTED)

- **인트라데이(분봉)**
  - **Intraday:** GET https://www.alphavantage.co/query? function=TIME_SERIES_INTRADAY
    &symbol=IBM&interval=5min&outputsize=full&datatype=json&apikey=YOUR_KEY

파라미터: interval=1min|5min|15min|30min|60min, outputsize=compact|full, datatype, apikey
(실시간/15분 지연 데이터는 **프리미엄** 필요. 기본은 종가 후 업데이트)

Alpha Vantage: 키 발급 · 사용

(B) 보조 엔드포인트

- **현재가(경량):** function=GLOBAL_QUOTE&symbol=IBM (현재가·변동률 등)
  - 문서 내 “Quote Endpoint” 섹션 참조

- **심볼 검색:** function=SYMBOL_SEARCH&keywords=tesla (심볼/이름 검색)
  - 문서 내 “Ticker Search Utility” 섹션

(C) 펀더멘털(재무)

- **회사 개요:** function=OVERVIEW&symbol=IBM (섹터/시총/배당 등)

- **재무제표:** function=INCOME_STATEMENT|BALANCE_SHEET|CASH_FLOW&symbol=IBM

  - 연/분기 제공
  - 문서의 **Fundamental Data** 섹션에 표기.

Alpha Vantage: 키 발급 · 사용

(D) 환율 · 가상자산

- **실시간 환율:**
  - function=CURRENCY_EXCHANGE_RATE&from_currency=USD&to_currency=KRW

- **FX 일/주/월봉:**
  - function=FX_DAILY|FX_WEEKLY|FX_MONTHLY&from_symbol=USD&to_symbol=KRW (Forex 섹션)

- **암호화폐 시계열:**
  - function=DIGITAL_CURRENCY_DAILY&symbol=BTC&market=USD (Crypto 섹션)

(E) 기술적 지표(예: SMA)

- **SMA:**
  function=SMA&symbol=IBM&interval=daily&time_period=20&series_type=close

Alpha Vantage: 키 발급 · 사용

```
import time, requests, pandas as pd

BASE = "https://www.alphavantage.co/query"

def av_get(params: dict, apikey: str, pause: float = 15.0):
    p = dict(params)
    p["apikey"] = apikey
    r = requests.get(BASE, params=p, timeout=30)
    r.raise_for_status()

   if p.get("datatype") == "csv":
        data = r.text
    else:
        data = r.json()

        if "Note" in data or "Information" in data:
            # 리밋 초과 등 메시지
            time.sleep(pause)
    return data
```

- Alpha Vantage 요청 공통 함수.
  - params: {'function': 'TIME_SERIES_DAILY', ...}
  - apikey: 발급 키
  - pause : 호출 사이 슬립(초) — 무료키 레이트리밋 회피용

- 반환: dict(JSON) 또는 CSV(str)

Yfinance : 키 불필요

- **장점: 키 없이 간단히 시세/배당/재무 접근, pandas 친화적.**
- **한국 종목: 티커에 거래소 접미사 사용(.KS=코스피, .KQ=코스닥).**
- **정합성: 분할/배당 반영(Adjusted) 여부 확인, 거래정지/상폐 히스토리 주의.**

A. Ticker 객체 메서드 (한 종목의 상세)

```
import yfinance as yf
t = yf.Ticker("AAPL")
```

- **가격 시계열**: t.history(...)
- **배당/분할**: t.dividends, t.splits, t.actions
- **재무제표**: t.income_stmt, t.balance_sheet, t.cashflow (+ 분기/TTM 버전)
- **실적/일정**: t.earnings_dates, t.calendar
- **리서치/홀더**: t.recommendations, t.institutional_holders 등

Yfinance : 키 불필요

B. yfinance.download() **(여러 종목의 시계열)**

```
yfinance.download(
  tickers, start=None, end=None, period=None, interval='1d',
  auto_adjust=None, back_adjust=False, actions=False,
  prepost=False, group_by='column', threads=True,
  ignore_tz=None, keepna=False, repair=False,
  rounding=False, timeout=10, session=None, multi_level_index=True
)
```

- 주요 파라미터
  - tickers: 티커(문자열/리스트)
  - period: 1d,5d,1mo,3mo,6mo,1y,2y,5y,10y,ytd,max (또는 start/end 직접 지정)
  - interval: 1m,2m,5m,15m,30m,60m,90m,1h,1d,5d,1wk,1mo,3mo *(분봉은 최근 60일 제한)*
  - auto_adjust: 배당/분할 반영 자동 조정 (기본 True)
  - actions: 배당/분할 데이터 동시 수집
  - 그 외 정렬/스레드/타임아웃 등 옵션 다수

네이버 금융

■ 공식 Open API가 없어 이용

- api.finance.naver.com/siseJson.naver의 비공식 JSON 엔드포인트
- pandas-datareader의 Naver 리더
- FinanceDataReader로 KRX 종목코드/시세 수집

네이버 금융

1) 비공식 엔드포인트: siseJson.naver

```
https://api.finance.naver.com/siseJson.naver
  ?symbol=005930
  &requestType=1
  &startTime=20140101
  &endTime=20250921
  &timeframe=day
```

- **symbol**: 6자리 종목코드(예: 삼성전자 005930)
- **requestType**: 0 또는 1 (공식 문서 없음 — 통상 0/1 둘 다 응답)
- **startTime / endTime**: YYYYMMDD 날짜 범위
- **timeframe**: day | week | month (일/주/월 봉)

이 엔드포인트는 **공식 문서가 없는 비공식**입니다.

```
import requests, pandas as pd, io, ast

def naver_ohlcv(symbol: str, start: str, end: str, timeframe: str = "day"):
    url = "https://api.finance.naver.com/siseJson.naver"
    params = dict(symbol=symbol, requestType=1, startTime=start, endTime=end, timeframe=timeframe)
    r = requests.get(url, params=params, timeout=30)
    r.raise_for_status()

    txt = r.text.strip()

    txt = txt[txt.find('['): txt.rfind(']')+1]
    data = ast.literal_eval(txt)

    if not data or len(data) < 2:
        return pd.DataFrame()

    cols = data[0]
    rows = data[1:]
    df = pd.DataFrame(rows, columns=cols)

    col_map = {
        '날짜':'date','시가':'open','고가':'high','저가':'low','종가':'close','거래량':'volume '       }
    df = df.rename(columns={c: col_map.get(c, c) for c in df.columns})

    num_cols = ['open','high','low','close','volume']
    for c in num_cols:
        if c in df.columns:
            df[c] = pd.to_numeric(df[c], errors='coerce')
    if 'date' in df.columns:
        df['date'] = pd.to_datetime(df['date'].str.replace('.', '-'), errors='coerce')

    return df.dropna(subset=['date']).sort_values('date').reset_index(drop=True)

    df = naver_ohlcv("005930", "20180101", "20250921", "day") # 사용 예: 삼성전자 2018~오늘, 일봉
    print(df.tail())
```

네이버 금융

2) pandas-datareader의 Naver 리더

```
import pandas_datareader as pdr
from pandas_datareader.naver import NaverDailyReader
import pandas as pd

symbol = "005930"  # 단일 심볼만 지원
start, end = pd.to_datetime("2018-01-01"), pd.to_datetime("2025-09-21")

df = NaverDailyReader(symbols=symbol, start=start, end=end).read()
print(df.tail())
```

네이버 금융

3) FinanceDataReader로 KRX 종목코드/시세 가져오기

```
# pip install finance-datareader
import FinanceDataReader as fdr

# 1) KRX 상장 종목 전체 (코드/이름/섹터 등)
krx = fdr.StockListing('KRX')
print(krx[['Symbol','Name']].head())

# 2) 단일 종목 시세
df = fdr.DataReader('005930', '2018-01-01', '2025-09-21')
print(df.tail())
```