[5-1. main.py파일 작성하기]

1. main.py 파일의 역할

- 프로그램의 시작점 :
  - 파이썬 프로젝트에서 'main.py' 파일은 주로 프로그램의 진입점(entry point)으로 사용되며, 프로그램이 실행될 때 가장 먼저 호출되는 코드를 포함합니다.
  - 이 파일에서 다른 모듈과 패키지를 불러오고, 필요한 초기 설정을 수행하며, 최상위 수준의 로직을 실행합니다.
- 명시적 실행 컨텍스트 제공 :
  - 'main.py'에서는 일반적으로 "if **name** == '**main**':" 이라는 구문을 사용하여 직접 실행될 때만 특정 코드가 실행되도록 할 수 있습니다.
  - 이 구문은 모듈이 다른 파일에 의해 import 될 때는 실행되지 않도록 보장합니다.
  - 이는 모듈을 재사용할 때 부작용을 방지해줍니다.
- 배포와 설치 :
  - 애플리케이션을 배포할 때 'main.py'는 종종 실행 파일 또는 진입점 역할을 하여 사용자가 프로그램을 쉽게 실행할 수 있도록 도와줍니다.

2. main.py 내용 작성

2-1. 주요 모듈 import 하기

In [ ]:
```
import pandas as pd
from openpyxl import Workbook, load_workbook

from connector import Connector
```

2-2. main.py 함수 작성

In [ ]:
```python
def main():
    print("Insert or select cashflow data.")
    print("Choose one.")
    print("1. Insert cashflow data(INSERT_FCHotel_cashflow.xlsx)")
    print("2. Select data")
    case_no = input(">>> ")

    if case_no == '1':
        insert_cashflow_data()
    elif case_no == '2':
        query = input("query > ")
        xlfilename = input("xlfilename > ")
        select_cashflow_data(query, xlfilename)
    else:
        print("Wrong Choice")
```

2-3. INSERT 함수 작성

In [ ]:
```python
def insert_cashflow_data():
    xlfilename = 'INSERT_FCHotel_cashflow.xlsx'

    # 엑셀 파일 로드
    workbook = load_workbook(xlfilename)

    # Insert Data가 있는 Sheet 설정
    sheet = workbook['InsertSheet']

    # 첫번째 행(컬럼 이름) 가져오기
    columns = [
        cell.value for cell
        in next(sheet.iter_rows(min_row=1, max_row=1))
    ]

    # 데이터를 딕셔너리 리스트로 변환
    data = []
    for row in sheet.iter_rows(min_row=2):
        record = {
            columns[i]: cell.value
            for i, cell in enumerate(row)
        }
        data.append(record)

    # 딕셔너리 리스트를 DataFrame으로 변환
    df = pd.DataFrame(data)

    with Connector() as conn:
        # INSERT 쿼리문 작성
        query = """
            INSERT INTO cashflow_tbl (
                account_id, dw_date, category, client,
                deposit, withdrawal, description
            )
            VALUES (%s, %s, %s, %s, %s, %s, %s);
        """

        # DataFrame으로부터 한줄씩 데이터를 읽어서 INSERT 쿼리문 실행
        for index, row in df.iterrows():
            val = (
                row['account_id'], row['dw_date'], row['category'],
                row['client'], row['deposit'], row['withdrawal'],
                row['description']
            )
            conn.cursor.execute(query, val)

        # 변경사항 커밋
        conn.connection.commit()
```

2-4. SELECT 함수 작성

In [ ]:
```python
def select_cashflow_data(query, xlfilename):
    with Connector() as conn:
        conn.cursor.execute(query)
        result = conn.cursor.fetchall()

    # 새로운 워크북(엑셀 파일) 생성
    workbook = Workbook()

    # 활성 시트(첫번째 시트) 선택
    sheet = workbook.active

    # 데이터의 키를 엑셀 파일의 첫번째 행에 헤더로 삽입
    headers = result[0].keys()
    sheet.append(list(headers))

    # 각 딕셔너리의 값을 차례로 행에 삽입
    for item in result:
        sheet.append(list(item.values()))

    # 엑셀 파일 저장
    workbook.save(xlfilename)
```

3. main() 함수 테스트하기

In [ ]:
```python
main()
```

```
Insert or select cashflow data.
Choose one.
1. Insert cashflow data(INSERT_FCHotel_cashflow.xlsx)
2. Select data


   >>> 2
   query > SELECT * FROM cashflow.cashflow_tbl;
   xlfilename > data_select_main1.xlsx
```

4. main.py 파일 생성하기

1. main.py 파일 생성
2. 작성된 코드 붙여넣기
3. 아래 구문 추가

```
if __name__ == '__main__':
main()
```

1. main.py 파일 실행하기

In [ ]:
```
%run main.py
```

```
Insert or select cashflow data.
Choose one.
1. Insert cashflow data(INSERT_FCHotel_cashflow.xlsx)
2. Select data


   >>> 2

   query > SELECT account_id, SUM(deposit) AS total_deposit,
   SUM(withdrawal) AS total_withdrawal, SUM(deposit) -
   SUM(withdrawal) AS balance FROM cashflow_tbl GROUP BY
   account_id;

   xlfilename > data_select_main2.xlsx
```