Python Mini Project

직원 DataBase 관리

김경수

개요

- 회사 , 그룹 등의 관리를 위해 데이터를 저장하는 수단 중 , 가장 기본적인 , 그리고 필수적인 정보를 담고 있는 DB를 SQL과 Jupyter Notebook을 활용하여 구현

목차

1. 데이터베이스의 기본 틀을 SQL을 활용하여 생성
2. 생성한 데이터베이스를 Jupyter Notebook에 가져와 입력 , 수정 삭제 등의 작업 수행
3. 업로드 된 데이터베이스를 EXCEL 파일로 변환하여 저장

1. 데이터베이스의 기본 틀을 SQL을 활용하여 생성

```
Create table company(
사원번호 number(5) primary key,
이름 varchar2(10) not null,
나이 number(2) not null,
부서 varchar2(20),
직급 varchar2(10),
입사년도 date not null,
주소 varchar2(30),
월급 number(5) not null);

commit;
```

자료를 관리할 때 사원번호를 기준으로 관리할 것이기 때문에 사원번호를 Primary key로 설정

2.생성한 데이터베이스를 Jupyter Notebook에 가져와 입력 , 수정 삭제 등의 작업 수행(company.py)

```
def dbConnection():
    import cx_Oracle as cx
    conn = cx.connect
    return conn
```

Jupyter Notebook과 만든 테이블이 있는 SQL과 연동 후 conn 객체로 생성

•함수로 묶는 이유 : 작업을 끝낸 모듈을 불러와서 하는 도중 , main 실행 코드가 있는 부분과 모듈 부분의 변수의 충돌 방지를 위해 모듈에서 함수 정의 후 가져오기 위함

```
def main2():
    conn = dbConnection()
    cur = conn.cursor()
    while True:
        view()
        print()
        print('============menu===========')
        print('1.신규 사원 추가')
        print('2.사원 정보 수정')
        print('3.사원 정보 삭제')
        print('4.특정 사원 정보 보기')
        print('5.전체 사원 정보 보기')
        print('6.작업 종료')
        print('============================')
        print()
        try:
            menu = int(input('업무를 선택하세요 : '))
        except:
            print('숫자를 입력하세요.')
        else:
            if menu==1:
                add_employee()
            elif menu==2:
                correction()
            elif menu==3:
                dele()
            elif menu==4:
                employee_show()
            elif menu==5:
                show_all()
            elif menu==6:
                print('작업을 종료합니다')
                break
            else:
                print('잘못된 메뉴 선택입니다')
```

- 작업을 수행하기 위한 Main2 함수 구현

```
def view():
    sql = 'select 사원번호 from company order by 사원번호'
    cur.execute(sql)
    m = cur.fetchall()
    employee_num.clear()

    for i in m:
        employee_num.append(i[0])
```

•수정 , 삭제 및 특정 사원의 정보를 확인할 때 , 사원의 사원 번호가 company 테이블 안에 존재하는가를 확인하기 위해 employee_num 객체를 생성.

1. 신규 사원 등록

```
if menu==1:
    add_employee()
    print('등록 완료')
```

- 테이블의 모든 열의 정보를 입력받아 한 행으로 추가

```
업무를 선택하세요 : 1
사원번호 : 1
이름 : 홍길동
나이 : 23
부서 : 개발부
직급 : 사원
입사년도(yyyy.mm.dd 형식) : 2023.09.01
주소 : 서울시 금천구 가산동
월급(만원) : 250
등록 완료
```

```
def add_employee():
    try:
        emp_num = int(input('사원번호 : '))
        name = input('이름 : ')
        age = int(input('나이 : '))
        dept = input('부서(비워도 됨) : ')
        rank = input('직급(비워도 됨) : ')
        join_year = input('입사년도(yyyy.mm.dd 형식) : ')
        address = input('주소(비워도 됨) : ')
        salary = int(input('월급(만원) : '))

    except:
        print('정보를 올바르게 입력하세요. 초기 화면으로 돌아갑니다.')
    else:
        if emp_num in employee_num:
            print('사원번호는 중복될 수 없습니다.')
        else:
            sql = 'insert into company values(:1,:2,:3,:4,:5,:6,:7,:8)'
            cur.execute(sql,[emp_num , name , age , dept , rank , join_year , address , salary])

            conn.commit()
            print('등록 완료')
```

수정과 삭제

```
elif menu==2:
    correction()
elif menu==3:
    dele()
```

```
def correction():

    try:
        num = int(input('수정할 사원의 사원번호를 입력하세요 : '))
    except:
        print('사원번호는 숫자로 입력해야 합니다. 초기 화면으로 돌아갑니다.')
    else:
        if num not in employee_num:
            print('사원번호가 존재하지 않습니다')
        else:
            print('1.이름 2.나이 3.부서 4.직급 5.입사년도 6.주소 7.월급')
            try:
                num2 = int(input('어떤 정보를 수정할지 번호를 고르세요 : '))
            except:
                print('숫자를 입력하셔야 합니다. 초기 화면으로 돌아갑니다.')
            else:
                a = ['이름','나이','부서','직급','입사년도','주소','월급']

                b = a[num2-1]

                num3 = input('수정할 정보를 입력하세요 : ')

                if num2 == 2 or num2 == 7:
                    sql = f'UPDATE company SET {b} = :1 WHERE 사원번호 = :2'
                    num4 = int(num3)
                    cur.execute(sql,[num4,num])
                    conn.commit()
                    print('수정 완료')
                elif num2 == 6 or num2 ==1 or num2 == 3 or num2 == 4 or num2 == 5:
                    sql = f'UPDATE company SET {b} = :1 WHERE 사원번호 = :2'

                    cur.execute(sql,[num3,num])
                    conn.commit()
                    print('수정 완료')
                else:
                    print('수정할 정보가 옳지 않습니다')
```

```
업무를 선택하세요 : 2
수정할 사원의 사원번호를 입력하세요 : 1
1.이름 2.나이 3.부서 4.직급 5.입사년도 6.주소 7.월급
어떤 정보를 수정할지 번호를 고르세요 : 2
수정할 정보를 입력하세요 : 24
수정 완료
```

```
def dele():
    try:
        inp = int(input('삭제할 사원의 사원번호를 입력하세요 : '))
    except:
        print('숫자를 입력하셔야 합니다. 초기 화면으로 돌아갑니다.')
    else:
        if inp in employee_num:
            sql = 'delete from company where 사원번호 = :1'

            cur.execute(sql, [inp])
            conn.commit()
            print('삭제 완료')
        else:
            print('사원번호가 존재하지 않습니다')
```

- 수정 및 삭제할 사원의 사원 번호가 존재해야 하므로 사원 번호가 있을 경우와 없을 경우를 나눠서 수행

```
업무를 선택하세요 : 3
삭제할 사원의 사원번호를 입력하세요 : 1
삭제 완료
```

특정 사원 보기

```
elif menu==4:
    employee_show()
```

- 입력 받은 사원 번호가 존재하면 그 사원 번호에 해당하는 정보를 출력하기

```
def employee_show():
    try:
        inp2 = int(input('보고 싶은 사원의 사원번호를 입력하세요 : '))
    except:
        print('숫자를 입력해야 합니다. 초기 화면으로 돌아갑니다.')
    else:
        if inp2 not in employee_num:
            print('사원번호가 존재하지 않습니다')
        else:
            sql = 'select * from company where 사원번호 = :1'

            cur.execute(sql,[inp2])
            a = cur.fetchall()
            print(a)
```

```
업무를 선택하세요 : 4
보고 싶은 사원의 사원번호를 입력하세요 : 0
[(0, '강감찬', 59, '총괄', '대표', datetime.datetime(1990, 2, 1, 0, 0), '서울시 강남구', 2000)]
```

모든 사원 정보 보기

```
elif menu==5:
    show_all()
```

- 모든 사원의 정보를 보기 위해 저장된 테이블의 모든 정보 출력

```
def show_all():
    sql = 'select * from company'
    cur.execute(sql)
    m = cur.fetchall()
    for i in m:
        print(i)
```

```
업무를 선택하세요 : 5
(0, '강감찬', 59, '총괄', '대표', datetime.datetime(1990, 2, 1, 0, 0), '서울시 강남구', 2000)
(1, '홍길동', 23, '개발부', '사원', datetime.datetime(2023, 9, 1, 0, 0), '서울시 금천구 가산동', 250)
(2, '장보고', 35, '개발부', '팀장', datetime.datetime(2010, 4, 2, 0, 0), '경기도 안양시', 400)
(3, '김유신', 38, '영업부', '팀장', datetime.datetime(2008, 8, 4, 0, 0), '경기도 수원시', 500)
(4, '유관순', 25, '경리부', '사원', datetime.datetime(2021, 3, 2, 0, 0), '경기도 시흥시', 220)
(5, '이순신', 33, '경영부', '사원', datetime.datetime(2023, 3, 4, 0, 0), '서울시 금천구 독산동', 220)
(6, '김경수', 28, None, None, datetime.datetime(2023, 9, 6, 0, 0), None, 250)
```

작업 종료

```
elif menu==6:
    print('작업을 종료합니다')
    break
```

```
cur.close()
conn.commit()
conn.close()
```

- 작업 종료 후 , 메모리 낭비 및 충돌을 방지하기 위해 DB와의 연동 해제

3.업로드 된 데이터베이스를 EXCEL 파일로 변환하여 저장

```
import pandas as pd
```

```
import cx_Oracle as cx
conn = cx.connect('scott' , '1234','127.0.0.1:1521/xe')
cur = conn.cursor()

sql = 'select * from company order by 사원번호'
cur.execute(sql)

rs = cur.fetchall()

df = pd.DataFrame(rs , columns=[i[0] for i in cur.description])
df.to_excel('output.xlsx',encoding = 'UTF-8' , index=False)
```

Cur.description : 열의 정보 추출
Fetchall : 행의 정보 추출

데이터프레임으로 변환 후 저장하기 위해 Pandas 패키지 불러오기.

Sql문을 작성 후 안에 있는 모든 요소를 fetchall로 가져온 후 , 데이터프레임 생성

사원번호	이름	나이	부서	직급	입사년도	주소	월급
0	강감찬	59	총괄	대표	1990년 2월 1일	서울시 강남구	2000
1	홍길동	23	개발부	사원	2023년 9월 1일	서울시 금천구 가산동	250
2	장보고	35	개발부	팀장	2010년 4월 2일	경기도 안양시	400
3	김유신	38	영업부	팀장	2008년 8월 4일	경기도 수원시	500
4	유관순	25	경리부	사원	2021년 3월 2일	경기도 시흥시	220
5	이순신	33	경영부	사원	2023년 3월 4일	서울시 금천구 독산동	220
6	김경수	28			2023년 9월 6일		250

감사합니다.