Week 1 Practice - Dipak Bange

You will need:

- Chapter 1 (SQL Cook Book). In this notebook you will be practicing the code provided in the chapter.
- Download emp.csv and dept.csv from the canvas Week 1 Practice

---

- Step 1-4: You will create a database week1.db
- Step 5: Practice Chapter 1 code
- Step 6: Close db connection
- Step 7: Open db connection using week1.db (you do not need step1-4 aanymore)

```
import sqlite3
import pandas as pd
```

STEP 1. Create a database named week1. You shsold have a new file week1.db in your local directory.

```
conn = sqlite3.connect('week1.db')
c = conn.cursor()
```

STEP 2. Read emp.csv and create a table emp

```
read_emp = pd.read_csv(r'emp.csv')
read_emp.to_sql('emp', conn, if_exists='append', index = False) # Insert the values from the csv file into the table 'emp'
```

STEP 3. read dept.csv and create a table dept

```
read_dept = pd.read_csv(r'dept.csv')
read_dept.to_sql('dept', conn, if_exists='append', index = False) # Insert the values from the csv file into the table 'dept'
```

Execution Examples

SQL statements will be executed with

c.execute(''' SQL code ''')

```
#Example 1
for row in c.execute('''
select * from emp
'''):
    print(row)
```

```
(7369, 'SMITH', 'CLERK', 7902.0, '17-Dec-05', 800, None, 20)
(7499, 'ALLEN', 'SALESMAN', 7698.0, '20-Feb-06', 1600, 300.0, 30)
(7521, 'WARD', 'SALESMAN', 7698.0, '22-Feb-06', 1250, 500.0, 30)
(7566, 'JONES', 'MANAGER', 7839.0, '2-Apr-06', 2975, None, 20)
(7654, 'MARTIN', 'SALESMAN', 7698.0, '28-Sep-06', 1250, 1400.0, 30)
(7698, 'BLAKE', 'MANAGER', 7839.0, '1-May-06', 2850, None, 30)
(7782, 'CLARK', 'MANAGER', 7839.0, '9-Jun-06', 2450, None, 10)
(7788, 'SCOTT', 'ANALYST', 7566.0, '9-Dec-07', 3000, None, 20)
(7839, 'KING', 'PRESIDENT', None, '17-Nov-06', 5000, None, 10)
(7844, 'TURNER', 'SALESMAN', 7698.0, '8-Sep-06', 1500, 0.0, 30)
(7876, 'ADAMS', 'CLERK', 7788.0, '12-Jan-08', 1100, None, 20)
(7900, 'JAMES', 'CLERK', 7698.0, '3-Dec-06', 950, None, 30)
(7902, 'FORD', 'ANALYST', 7566.0, '3-Dec-06', 3000, None, 20)
(7934, 'MILLER', 'CLERK', 7782.0, '23-Jan-07', 1300, None, 10)
```

```
colnames = c.description
for row in colnames:
    print(row[0])
```

```
EMPNO
ENAME
JOB
MGR
```

```
HIREDATE
SAL
COMM
DEPTNO
```

To print a table, use fetchall() to collect data and add column names thaht you have selected.

```
# Example 2
c.execute('''
select * from emp
''')

df = pd.DataFrame(c.fetchall(), columns=['EMPNO',
'ENAME',
'JOB',
'MGR',
'HIREDATE',
'SAL',
'COMM',
'DEPTNO'])
print(df)
```

	EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
0	7369	SMITH	CLERK	7902.0	17-Dec-05	800	NaN	20
1	7499	ALLEN	SALESMAN	7698.0	20-Feb-06	1600	300.0	30
2	7521	WARD	SALESMAN	7698.0	22-Feb-06	1250	500.0	30
3	7566	JONES	MANAGER	7839.0	2-Apr-06	2975	NaN	20
4	7654	MARTIN	SALESMAN	7698.0	28-Sep-06	1250	1400.0	30
5	7698	BLAKE	MANAGER	7839.0	1-May-06	2850	NaN	30
6	7782	CLARK	MANAGER	7839.0	9-Jun-06	2450	NaN	10
7	7788	SCOTT	ANALYST	7566.0	9-Dec-07	3000	NaN	20
8	7839	KING	PRESIDENT	NaN	17-Nov-06	5000	NaN	10
9	7844	TURNER	SALESMAN	7698.0	8-Sep-06	1500	0.0	30
10	7876	ADAMS	CLERK	7788.0	12-Jan-08	1100	NaN	20
11	7900	JAMES	CLERK	7698.0	3-Dec-06	950	NaN	30
12	7902	FORD	ANALYST	7566.0	3-Dec-06	3000	NaN	20
13	7934	MILLER	CLERK	7782.0	23-Jan-07	1300	NaN	10

Basics of SQL Queries

SELECT: Statement used to select rows and columns from a database.

FROM: Specifies which table in the database you want to direct your query to.

WHERE: Clause for filtering for specified value(s).

GROUP BY: Aggregating data. Needs to be used in conjunction with SQL aggregating functions like `SUM` and `COUNT`.

ORDER BY: Sorting columns in the database.

JOIN: Joins are used to combine tables with one another.

UNION, **INTERSECT/EXCEPT**: Set operations. Unioning in SQL allows one to append tables on top of one another.

Step 5. Practice Chapter 1

```
for row in c.execute('''select * from emp'''):
    print(row)
```

```
(7369, 'SMITH', 'CLERK', 7902.0, '17-Dec-05', 800, None, 20)
(7499, 'ALLEN', 'SALESMAN', 7698.0, '20-Feb-06', 1600, 300.0, 30)
(7521, 'WARD', 'SALESMAN', 7698.0, '22-Feb-06', 1250, 500.0, 30)
(7566, 'JONES', 'MANAGER', 7839.0, '2-Apr-06', 2975, None, 20)
(7654, 'MARTIN', 'SALESMAN', 7698.0, '28-Sep-06', 1250, 1400.0, 30)
(7698, 'BLAKE', 'MANAGER', 7839.0, '1-May-06', 2850, None, 30)
(7782, 'CLARK', 'MANAGER', 7839.0, '9-Jun-06', 2450, None, 10)
(7788, 'SCOTT', 'ANALYST', 7566.0, '9-Dec-07', 3000, None, 20)
(7839, 'KING', 'PRESIDENT', None, '17-Nov-06', 5000, None, 10)
(7844, 'TURNER', 'SALESMAN', 7698.0, '8-Sep-06', 1500, 0.0, 30)
(7876, 'ADAMS', 'CLERK', 7788.0, '12-Jan-08', 1100, None, 20)
(7900, 'JAMES', 'CLERK', 7698.0, '3-Dec-06', 950, None, 30)
(7902, 'FORD', 'ANALYST', 7566.0, '3-Dec-06', 3000, None, 20)
(7934, 'MILLER', 'CLERK', 7782.0, '23-Jan-07', 1300, None, 10)
```

```
for row in c.execute('''select * from emp where deptno=30'''):
    print(row)
```

```
(7499, 'ALLEN', 'SALESMAN', 7698.0, '20-Feb-06', 1600, 300.0, 30)
(7521, 'WARD', 'SALESMAN', 7698.0, '22-Feb-06', 1250, 500.0, 30)
(7654, 'MARTIN', 'SALESMAN', 7698.0, '28-Sep-06', 1250, 1400.0, 30)
(7698, 'BLAKE', 'MANAGER', 7839.0, '1-May-06', 2850, None, 30)
(7844, 'TURNER', 'SALESMAN', 7698.0, '8-Sep-06', 1500, 0.0, 30)
(7900, 'JAMES', 'CLERK', 7698.0, '3-Dec-06', 950, None, 30)
```

```
c.execute('''select job, sum(SAL) from emp group by job''')
columns_names = [description[0] for description in c.description]
display(pd.DataFrame(c.fetchall(), columns=columns_names))
```

1 to 5 of 5 entries Filter

index	JOB	sum(SAL)
0	ANALYST	6000
1	CLERK	4150
2	MANAGER	8275
3	PRESIDENT	5000
4	SALESMAN	5600

Show 25 per page

Like what you see? Visit the data table notebook to learn more about interactive tables.

```
c.execute('''select job, sum(SAL) as SAL from emp group by job having sum(SAL) > 5000 order by sum(SAL) DESC''')
columns_names = [description[0] for description in c.description]
display(pd.DataFrame(c.fetchall(), columns=columns_names))
```

	JOB	SAL
0	MANAGER	8275
1	ANALYST	6000
2	SALESMAN	5600

```
c.execute('''select job, count(1) as SAL from emp group by job order by count(1) DESC''')
columns_names = [description[0] for description in c.description]
display(pd.DataFrame(c.fetchall(), columns=columns_names))
```

	JOB	SAL
0	CLERK	4
1	SALESMAN	4
2	MANAGER	3
3	ANALYST	2
4	PRESIDENT	1

```
c.execute('''select * from dept''')
columns_names = [description[0] for description in c.description]
display(pd.DataFrame(c.fetchall(), columns=columns_names))
```

	DEPTNO	DNAME	LOC
0	10	ACCOUNTING	NEW YORK
1	20	RESEARCH	DALLAS
2	30	SALES	CHICAGO
3	40	OPERATIONS	BOSTON

```
c.execute('''
select e.*, d.dname, d.loc from emp e
left join dept d
on e.deptno = d.deptno
''')
columns_names = [description[0] for description in c.description]
display(pd.DataFrame(c.fetchall(), columns=columns_names))
```

	EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO	DNAME
0	7369	SMITH	CLERK	7902.0	17-Dec-05	800	NaN	20	RESEARCH
1	7499	ALLEN	SALESMAN	7698.0	20-Feb-06	1600	300.0	30	SALES
2	7521	WARD	SALESMAN	7698.0	22-Feb-06	1250	500.0	30	SALES
3	7566	JONES	MANAGER	7839.0	2-Apr-06	2975	NaN	20	RESEARCH
4	7654	MARTIN	SALESMAN	7698.0	28-Sep-06	1250	1400.0	30	SALES
5	7698	BLAKE	MANAGER	7839.0	1-May-06	2850	NaN	30	SALES
6	7782	CLARK	MANAGER	7839.0	9-Jun-06	2450	NaN	10	ACCOUNTING
7	7788	SCOTT	ANALYST	7566.0	9-Dec-07	3000	NaN	20	RESEARCH
8	7839	KING	PRESIDENT	NaN	17-Nov-06	5000	NaN	10	ACCOUNTING
9	7844	TURNER	SALESMAN	7698.0	8-Sep-06	1500	0.0	30	SALES

Step 6. Close the connection

```
conn.close()
```

Step 7. Open connection with your database week1.db

```
conn = sqlite3.connect('week1.db')
c = conn.cursor()


c.execute('''
select e.*, d.dname, d.loc from emp e
left join dept d
on e.deptno = d.deptno where e.deptno <> 20
''')
columns_names = [description[0] for description in c.description]
display(pd.DataFrame(c.fetchall(), columns=columns_names))
```

	EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO	DNAME	
0	7499	ALLEN	SALESMAN	7698.0	20-Feb-06	1600	300.0	30	SALES	CHIC/
1	7521	WARD	SALESMAN	7698.0	22-Feb-06	1250	500.0	30	SALES	CHIC/
2	7654	MARTIN	SALESMAN	7698.0	28-Sep-06	1250	1400.0	30	SALES	CHIC/
3	7698	BLAKE	MANAGER	7839.0	1-May-06	2850	NaN	30	SALES	CHIC/
4	7782	CLARK	MANAGER	7839.0	9-Jun-06	2450	NaN	10	ACCOUNTING	N Y(
5	7839	KING	PRESIDENT	NaN	17-Nov-06	5000	NaN	10	ACCOUNTING	N Y(

```
conn.close()
```

✓ 0s completed at 5:08 PM