Homework 1 Part 2: SQL

To start with, if you felt the class was unclear, check out the following tutorial: https://mode.com/sql-tutorial/introduction-to-sql/

Now! We'll be using sqlite to access a database. Start by downloading the sql lite file and putting it in the same directory as this notebook: https://www.kaggle.com/datasets/kaggle/sf-salaries (hit the 'download' button in the upper right). Check out the description of the data so you know the table / column names.

The following code will use sqlite to create a database connection.

```
import sqlite3
import pandas as pd

conn = sqlite3.connect("database.sqlite")
crsr = conn.cursor()
```

Before we proceed, please note that every task **must be completed using a single SQL query**, unless mentioned otherwise or given as prompt. (e.g. using print statements are fine, but you should not be using pandas library to work with the dataset.) The query should be stored in the query variable, like so:

query = 'your query here'

Exploration

Problem 1:

Try to create a query that gives you a data frame of the EmployeeName, JobTitle, and BasePay from the salaries table.

```
query = 'select EmployeeName, JobTitle, BasePay from Salaries'
df = pd.read_sql(query, conn)
df.head()
```

	EmployeeName	JobTitle	BasePay
0	NATHANIEL FORD	GENERAL MANAGER-METROPOLITAN TRANSIT AUTHORITY	167411.18
1	GARY JIMENEZ	CAPTAIN III (POLICE DEPARTMENT)	155966.02
2	ALBERT PARDINI	CAPTAIN III (POLICE DEPARTMENT)	212739.13
3	CHRISTOPHER CHONG	WIRE ROPE CABLE MAINTENANCE MECHANIC	77916
4	PATRICK GARDNER	DEPUTY CHIEF OF DEPARTMENT,(FIRE DEPARTMENT)	134401.6

Problem 2.

Modify your query from Problem 1 to limit it to the year 2012.

```
query = 'select EmployeeName, JobTitle, BasePay from Salaries where Year = 2012'

df = pd.read_sql(query, conn)
df.head()
```

	EmployeeName	JobTitle	BasePay
0	Gary Altenberg	Lieutenant, Fire Suppression	128808.87
1	Gregory Suhr	Chief of Police	302578.00
2	Khoa Trinh	Electronic Maintenance Tech	111921.00
3	Joanne Hayes-White	Chief, Fire Department	296943.01
4	Frederick Binkley	EMT/Paramedic/Firefighter	126863.19

Next steps: Generate code with df | View recommended plots | New interactive sheet

Problem 3:

Further limit the table to the year 2012, employees making under 150,000, and sort in descending order by salary.

```
query = 'select EmployeeName, JobTitle, BasePay from Salaries where Year = 2012 and TotalPay < 150000 order by BasePay desc'

df = pd.read_sql(query, conn)
df.head()
```

	EmployeeName	JobTitle	BasePay
0	Julie Kirschbaum	Project Manager 3	149881.06
1	Eric Jamison	Senior Physician Specialist	149746.72
2	Avantika Nath	Dentist	148539.61
3	Levis Owens	Nurse Practitioner	148404.45
4	Diana Strait	Attorney (Civil/Criminal)	148112.44

Next steps: Generate code with df View recommended plots New interactive sheet

Aggregation

Problem 4:

Get the average base pay from the table.

```
query = 'select avg(BasePay) from Salaries'

df = pd.read_sql(query, conn)
df.head()
```

	avg(BasePay)
0	66053.729288

Problem 5:

Produce and print the head of a dataframe that shows the average pay for each year (only use a single, simple query). Your result should have a column for the year and a column for the average base pay.

```
query = 'select Year, avg(BasePay) from Salaries group by Year'

df = pd.read_sql(query, conn)
df.head()
```

	Year	avg(BasePay)
0	2011	63595.956517
1	2012	65436.406857
2	2013	68509.832156
3	2014	66557.437750

Next steps: Generate code with df View recommended plots New interactive sheet

Problem 6:

Create a dataframe with averages of base pay, benefits, and overtime for each job title, as well as a column with the average of these three values.

```
query = """
select JobTitle, avg(BasePay), avg(benefits), avg(overtimePay),
((avg(BasePay) + avg(benefits) + avg(overtimePay))/3 )
from Salaries group by JobTitle
```

```
"""
df = pd.read_sql(query, conn)
```

Table Creation

Problem 7:

Now we'll create our own table in our database. Separate the Salaries table by Year, and add it back to the database. (You may use basic python to complete the task, however, should still use SQL to query the data.)

```
for y in ['2011','2012','2013','2014']:
    query = f"select * from Salaries where year = {y}"
    df = pd.read_sql(query, conn)
    df.to_sql(name='Y'+y, con=conn, if_exists='replace')
```

Table Joining

Problem 8:

We'll move on to a new dataset for the next steps. Download the dataset from here (https://www.kaggle.com/datasets/luizpaulodeoliveira/imdb-project-sql) and load the sqlite file same as before. Start by just selecting everything in the "movies" table to see what it looks like.

```
conn = sqlite3.connect("movies.sqlite")
query = 'select * from movies'

df = pd.read_sql(query, conn)
df.head()
```

	id	original_title	budget	popularity	release_date	revenue	title	vote_average	vote_count	overview	tagline
0	43597	Avatar	237000000	150	2009-12-10	2787965087	Avatar	7.2	11800	In the 22nd century, a paraplegic Marine is di...	Enter the World of Pandora.
1	43598	Pirates of the Caribbean: At World's End	300000000	139	2007-05-19	961000000	Pirates of the Caribbean: At World's End	6.9	4500	Captain Barbossa, long believed to be dead, ha...	At the end of the world, the adventure begins.
2	43599	Spectre	245000000	107	2015-10-26	880674609	Spectre	6.3	4466	A cryptic message from Bond's past sends him o...	A Plan No One Escapes
3	43600	The Dark Knight Rises	250000000	112	2012-07-16	1084939099	The Dark Knight Rises	7.6	9106	Following the death of District Attorney Harve...	The Legend Ends

Next steps: Generate code with df View recommended plots New interactive sheet

Problem 9:

Create a dataframe that includes the entire contents of "movies" table as well as the director's name.

```
query = 'select m.*, d.name from movies as m join directors as d on m.director_id = d.id'
df = pd.read_sql(query, conn)
```

Analysis

The next few problems will be more involved! You'll need to combine some concepts you've learned. For each cell, show your work. Remember, the answers should be in **a single query**.

Problem 10:

What is the average budget used for the top 10 grossing movies?

```
# query orders movies by revenue desc, limited to top 10
query = 'select * from movies order by revenue desc limit 10'
df = pd.read_sql(query, conn)
df.head(10)

# query gets avg(budget) from above query table
query = 'select avg(budget) from (select * from movies order by revenue desc limit 10)'
df = pd.read_sql(query, conn)
df.head()
```

	avg(budget)
0	195100000.0

Problem 11:

Which directors have the highest overall voting average? - show the top 5 directors' name and their average rating

```
# query from q9, which add director name to movies table
query = 'select m.*, d.name from movies as m join directors as d on m.director_id = d.id'
df = pd.read_sql(query, conn)
df.head(10)

# query uses query above to select director name and avg(vote_average)
# and group by directors and order by highest vote_average
# and limited to 5 entries
query = """
select name, avg(vote_average) as vote_avg
from (select m.*, d.name from movies as m join directors as d on m.director_id = d.id)
group by name
order by vote_avg desc
limit 5
"""
df = pd.read_sql(query, conn)
df.head()
```

	name	vote_avg
0	Gary Sinyor	10.00
1	Rohit Jugraj	9.50
2	Lance Hool	9.30
3	Floyd Mutrux	8.50
4	Tim McCanlies	8.45

Next steps: Generate code with df View recommended plots New interactive sheet

Problem 12:

What are the top five directors by how much their average budget is?

```
# query from q9, which add director name to movies table
query = 'select m.*, d.name from movies as m join directors as d on m.director_id = d.id'
df = pd.read_sql(query, conn)
df.head(10)

# query uses query above to select director name and avg(budget)
# and group by directors and order by highest average budget
# and limited to 5 entries
query = """
```

```
select name, avg(budget) as bud_avg
from (select m.*, d.name from movies as m join directors as d on m.director_id = d.id)
group by name
order by bud_avg desc
limit 5
"""
df = pd.read_sql(query, conn)
df.head(10)
```

	name	bud_avg
0	Byron Howard	2.600000e+08
1	Lee Unkrich	2.000000e+08
2	Dan Scanlon	2.000000e+08
3	David Yates	1.933333e+08
4	Brenda Chapman	1.850000e+08

Next steps: Generate code with df View recommended plots New interactive sheet