SELECT LAB

NASA has set its sights on Mars! To prepare for the mission and design their rocket, NASA needs to collect detailed information about each planet in the Solar System. In this lab, you'll gain experience querying a database using various SELECT statements, including selecting specific columns and applying SQL clauses like WHERE to retrieve the needed data.

Planets

Goals:

- Connect to a SQL database using Python
- Retrieve all information from a SQL table
- Retrieve a subset of records from a table using a `WHERE` clause
- Write SQL queries to filter and order results
- Retrieve a subset of columns from a table

Connecting to the Database

1. Import `sqlite3` as well as `pandas` for conveniently displaying results. Then, connect to the SQLite database located at `planets.db`.

In [1]:

```python
import pandas as pd # importing pandas
import sqlite3
import re  # For removing extra spaces

# connect database
conn = sqlite3.connect('planets.db')
# creating a cursor object
cur = conn.cursor()
```

Database Schema

2. write a code that shows Database Schema

In [2]:

```python
cur.execute("""SELECT sql FROM sqlite_master""")
planets_schema = cur.fetchall()
planets_schema

# Clean the schema by removing newlines and extra spaces
clean_schema = [re.sub(r'\s+', ' ', schema[0].replace('\n', ' ')).strip() for schem

for table_schema in clean_schema:
    print(table_schema)
```

```
CREATE TABLE planets (id INTEGER PRIMARY KEY, name TEXT, color TEXT, num_of_moons IN
TEGER, mass REAL, rings BOOLEAN)
```

```
In [3]: planets = pd.read_sql(""" SELECT * FROM planets;""", conn)
        planets
```

Out[3]:

	id	name	color	num_of_moons	mass	rings
0	1	Mercury	gray	0	0.55	0
1	2	Venus	yellow	0	0.82	0
2	3	Earth	blue	1	1.00	0
3	4	Mars	red	2	0.11	0
4	5	Jupiter	orange	68	317.90	0
5	6	Saturn	hazel	62	95.19	1
6	7	Uranus	light blue	27	14.54	1
7	8	Neptune	dark blue	14	17.15	1

SELECT

3.Select just the name and color of each planet

```
In [4]: name_color = pd.read_sql("""SELECT name, color FROM planets;""", conn)
        name_color
```

Out[4]:

	name	color
0	Mercury	gray
1	Venus	yellow
2	Earth	blue
3	Mars	red
4	Jupiter	orange
5	Saturn	hazel
6	Uranus	light blue
7	Neptune	dark blue

4. Select all columns for each planet whose mass is greater than 1.00

```
In [5]: planets_mass_greater_than_1 = pd.read_sql("""SELECT * FROM planets WHERE mass > 1.0
        planets_mass_greater_than_1
```

Out[5]:

	id	name	color	num_of_moons	mass	rings
0	5	Jupiter	orange	68	317.90	0
1	6	Saturn	hazel	62	95.19	1
2	7	Uranus	light blue	27	14.54	1
3	8	Neptune	dark blue	14	17.15	1

5. Select the name and color of each planet that has more than 10 moons

In [6]:

```
planets_more_than_10moons = pd.read_sql("""SELECT name, color FROM planets WHERE nu
planets_more_than_10moons
```

Out[6]:

	name	color
0	Jupiter	orange
1	Saturn	hazel
2	Uranus	light blue
3	Neptune	dark blue

6. Select the planet that has at least one moon and a mass less than 1.00

In [7]:

```
planet_with_atleast_one_moon_and_less_than_1mass = pd.read_sql("""SELECT * FROM pla
planet_with_atleast_one_moon_and_less_than_1mass
```

Out[7]:

	id	name	color	num_of_moons	mass	rings
0	4	Mars	red	2	0.11	0

7. Select the name and color of planets that have a color of blue, light blue, or dark blue

In [8]:

```
planets_blue_lightBlue_darkBlue = pd.read_sql("""SELECT name, color FROM planets WH
planets_blue_lightBlue_darkBlue
```

Out[8]:

	name	color
0	Earth	blue
1	Uranus	light blue
2	Neptune	dark blue

8. Select the name, color, and number of moons for the 4 largest planets that don't have rings and order them from largest to smallest

In [9]:
```
largest_planets = pd.read_sql(""" SELECT name, color, num_of_moons FROM planets WHE
largest_planets
```

Out[9]:

	name	color	num_of_moons
0	Jupiter	orange	68
1	Earth	blue	1
2	Venus	yellow	0
3	Mercury	gray	0

SQL Data Types