Dino Fun World Assignment

Jupyter Notebook Demonstration: Loading Data in Python ✓

Submission: Dino Fun World Assignment ✓

Jupyter Notebook Demonstration: Loading Data in Python

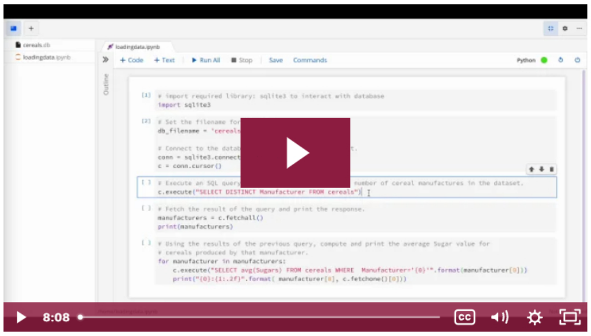
In the following video demonstration, you will see how to load data into a Jupyter Notebook to create a simple visualization. This Notebook loads a database of cereals and various properties about the cereals, and then computes and displays the average Sugar content (in grams) of each manufacturer's cereals.

Read the steps guiding you through the example code and learn each block's function.

Printable Directions

CSE 578_Reading_Jupyter Notebook Demonstration_ Loading Data in Python.pdf

Video - Jupyter Notebook Demonstration: Loading Data in Python



```
[1] # import required library: sqlite3 to interact with database
    import sqlite3

[2] # Set the filename for
    db_filename = 'cereals

    # Connect to the datab                                   t.
    conn = sqlite3.connect
    c = conn.cursor()

[ ] # Execute an SQL query                   number of cereal manufactures in the dataset.
    c.execute("SELECT DISTINCT Manufacturer FROM cereals")

[ ] # Fetch the result of the query and print the response.
    manufacturers = c.fetchall()
    print(manufacturers)

[ ] # Using the results of the previous query, compute and print the average Sugar value for
    # cereals produced by that manufacturer.
    for manufacturer in manufacturers:
        c.execute("SELECT avg(Sugars) FROM cereals WHERE  Manufacturer='{0}'".format(manufacturer[0]))
        print("{0}:{1:.2f}".format( manufacturer[0], c.fetchone()[0]))
```

Part 1: Import Library

Code:

PYTHON

```python
import sqlite3
```

Explanation:

The above single line of code includes the libraries needed for the rest of the example. In this simple example, you will need only include the sqlite3 library. Future examples and assignments will require additional libraries.

Part 2: Establish Database Name

Code:

PYTHON

```python
1 db_filename = 'cereals.db'
```

Explanation:

The above single line of code establishes the name of the database that contains the relevant data for the example. While this example uses a database containing cereal information here, your assignment will use a different database.

Part 3: Establish Connection

Code:

PYTHON

```python
conn = sqlite3.connect(db_filename)
c = conn.cursor()
```

Explanation:

The first line of code is called to establish a connection to the database. It takes a single argument i.e the database name, named in Part 2(line 1).

Once the connection to the database is established, the `conn.cursor()` method is called to create a cursor object associated with the database connection.

The cursor object is necessary to execute queries on the database.

Part 4: Simple Query

Code:

```
c.execute ("SELECT DISTINCT Manufacturer FROM cereals")
```

Explanation:

The above line of code performs a simple query on the database to return a list of all the manufacturers represented in the database. Note that while this line executes the query, it does not retrieve the results.

Part 5: Retrieve Results From Query

Code:

```
manufactures=c.fetchall()
print(manufactures)
```

Explanation:

The first line of code fetches all the results retrieved from previous query (Part 4). The print statement displays the results in a list format.

Part 6: Second Query

Code:

```
for manufacturer in manufacturers:
    c.execute("SELECT avg(Sugars) FROM cereals WHERE  Manufacturer='{0}'".format(ma
    print("{0}:{1:.2f}".format( manufacturer[0], c.fetchone()[0]))

```

Explanation:

In this block of code, the results of the previous query are iterated over in sending the second query, which returns the average Sugar value of the specified manufacturer. The print statement then displays the result by concatenating the manufacturer name [manufacturer[0]) and avg Sugar value using `c.fetchone()[0]' method.

Conclusion

This example demonstrates the process of loading a dataset and performing simple statistics on that dataset.