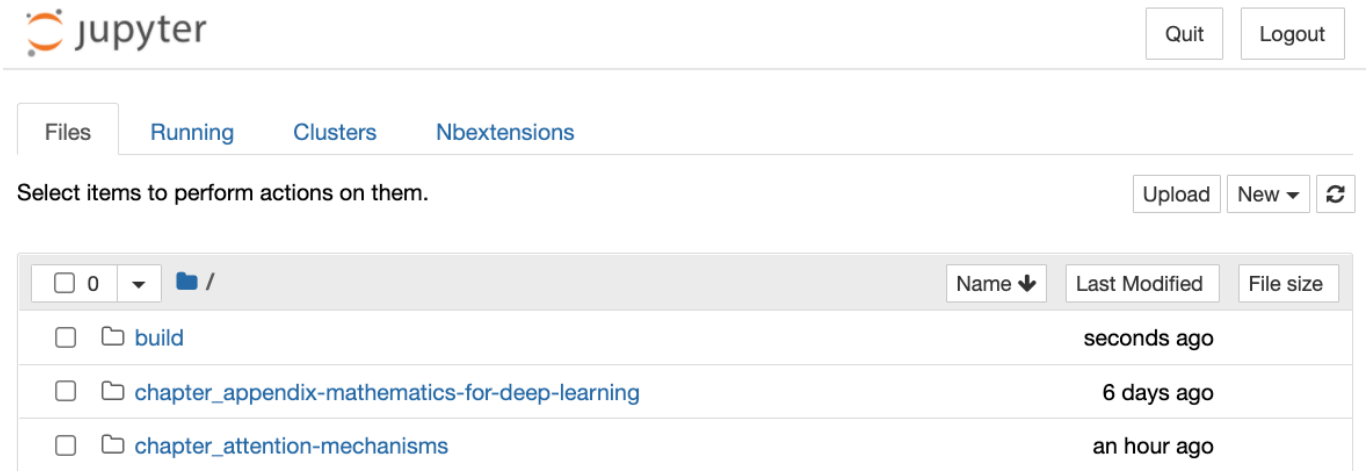
Using Jupyter Notebooks

:label: `sec_jupyter`

This section describes how to edit and run the code in each section of this book using the Jupyter Notebook. Make sure you have installed Jupyter and downloaded the code as described in :ref: `chap_installation` . If you want to know more about Jupyter see the excellent tutorial in their documentation.

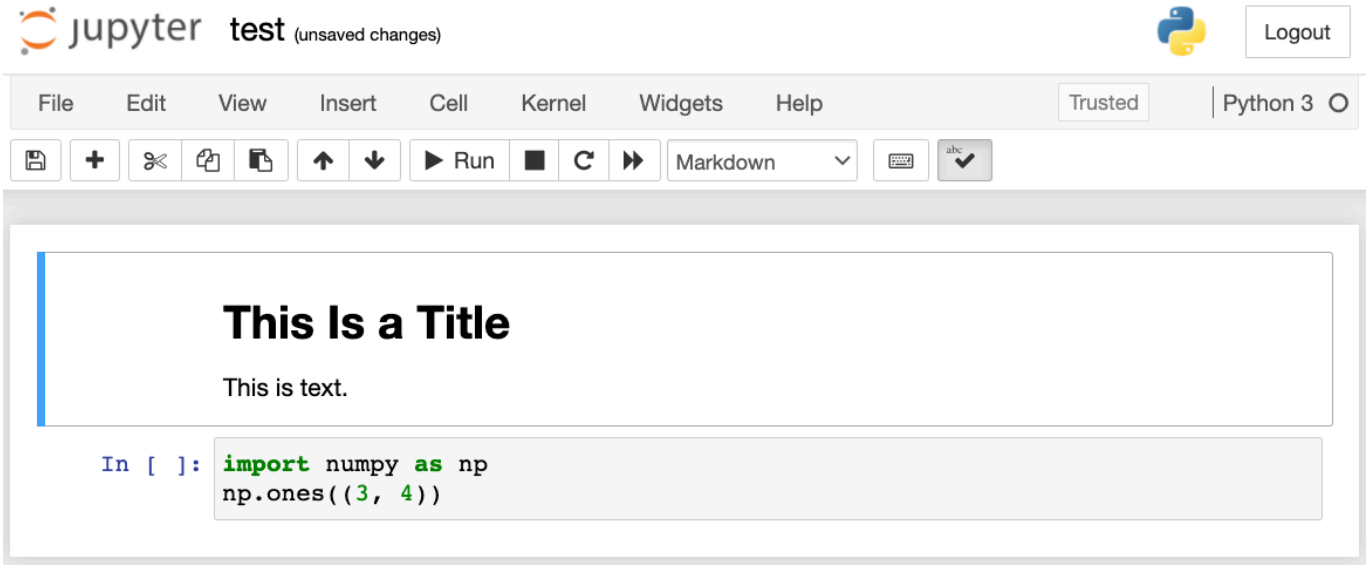
Editing and Running the Code Locally

Suppose that the local path of the book's code is `xx/yy/d2l-en/` . Use the shell to change the directory to this path (`cd xx/yy/d2l-en`) and run the command `jupyter notebook` . If your browser does not do this automatically, open http://localhost:8888 and you will see the interface of Jupyter and all the folders containing the code of the book, as shown in :numref: `fig_jupyter00` .



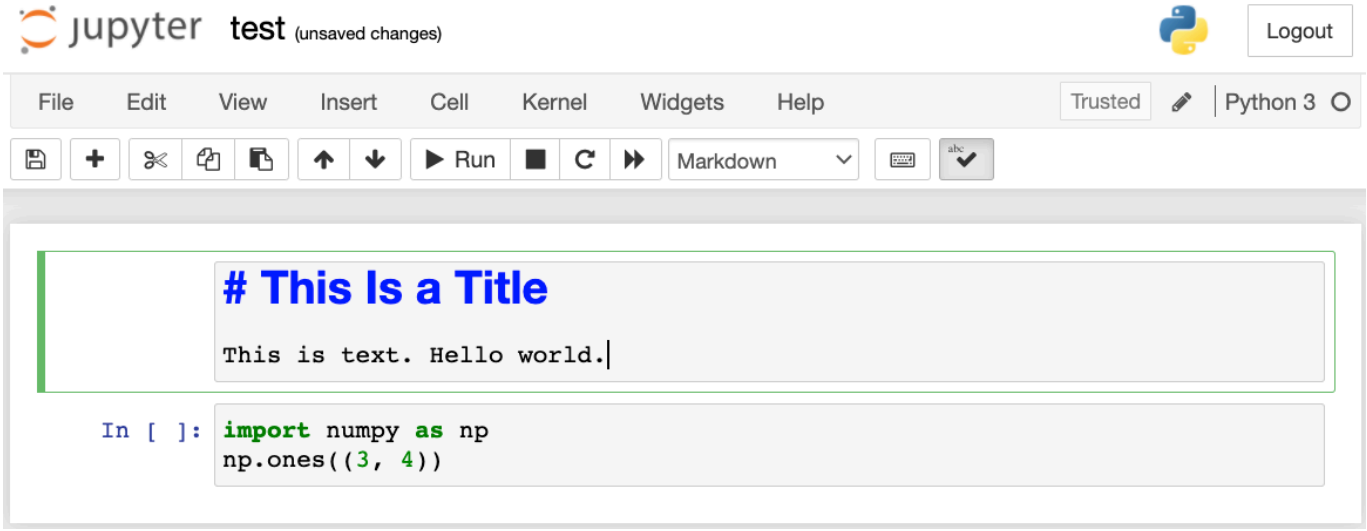
:width: `600px` :label: `fig_jupyter00`

You can access the notebook files by clicking on the folder displayed on the webpage. They usually have the suffix ".ipynb". For the sake of brevity, we create a temporary "test.ipynb" file. The content displayed after you click it is shown in :numref: `fig_jupyter01` . This notebook includes a markdown cell and a code cell. The content in the markdown cell includes "This Is a Title" and "This is text.". The code cell contains two lines of Python code.



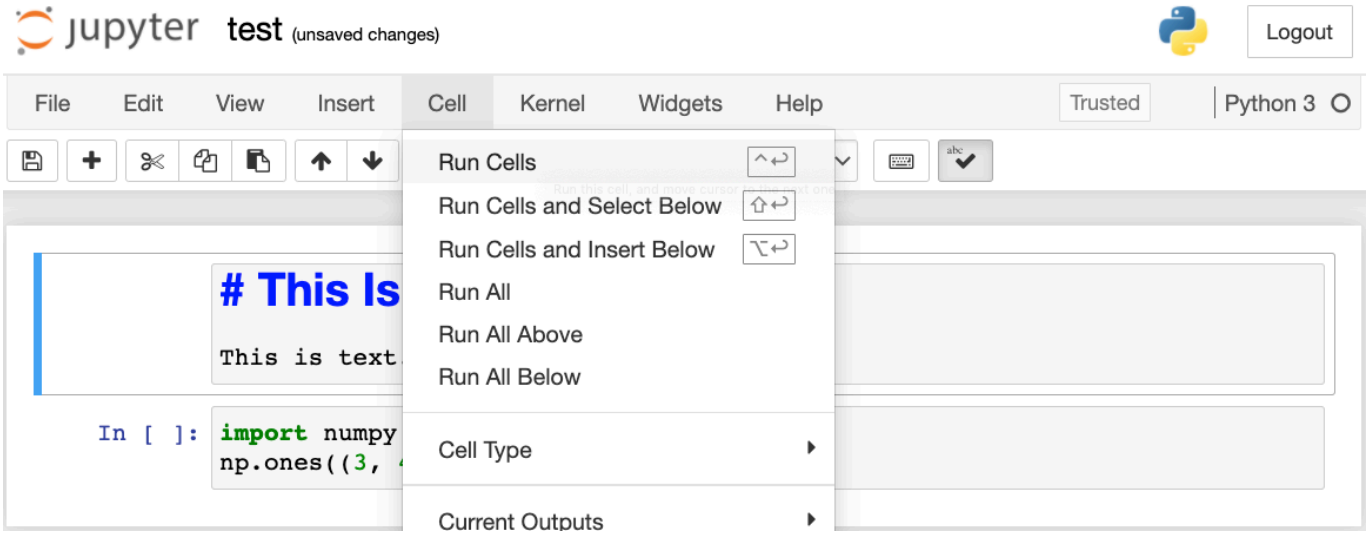
:width: `600px` :label: `fig_jupyter01`

Double click on the markdown cell to enter edit mode. Add a new text string "Hello world." at the end of the cell, as shown in :numref: `fig_jupyter02` .



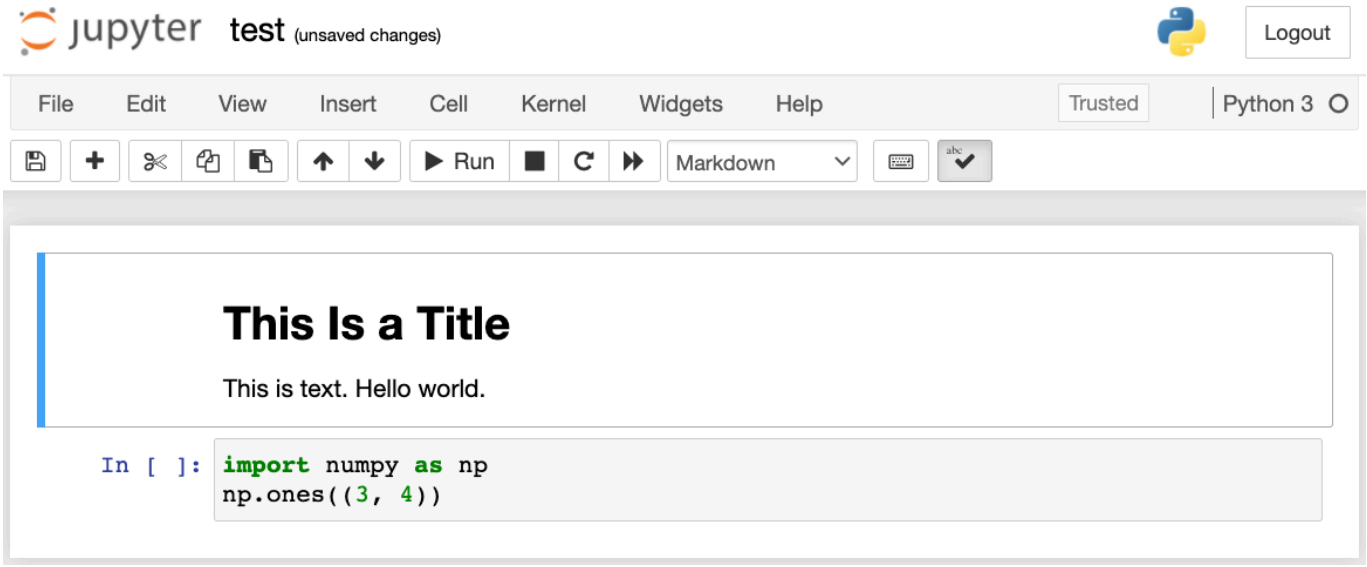
:width: `600px` :label: `fig_jupyter02`

As demonstrated in :numref: `fig_jupyter03` , click "Cell" → "Run Cells" in the menu bar to run the edited cell.



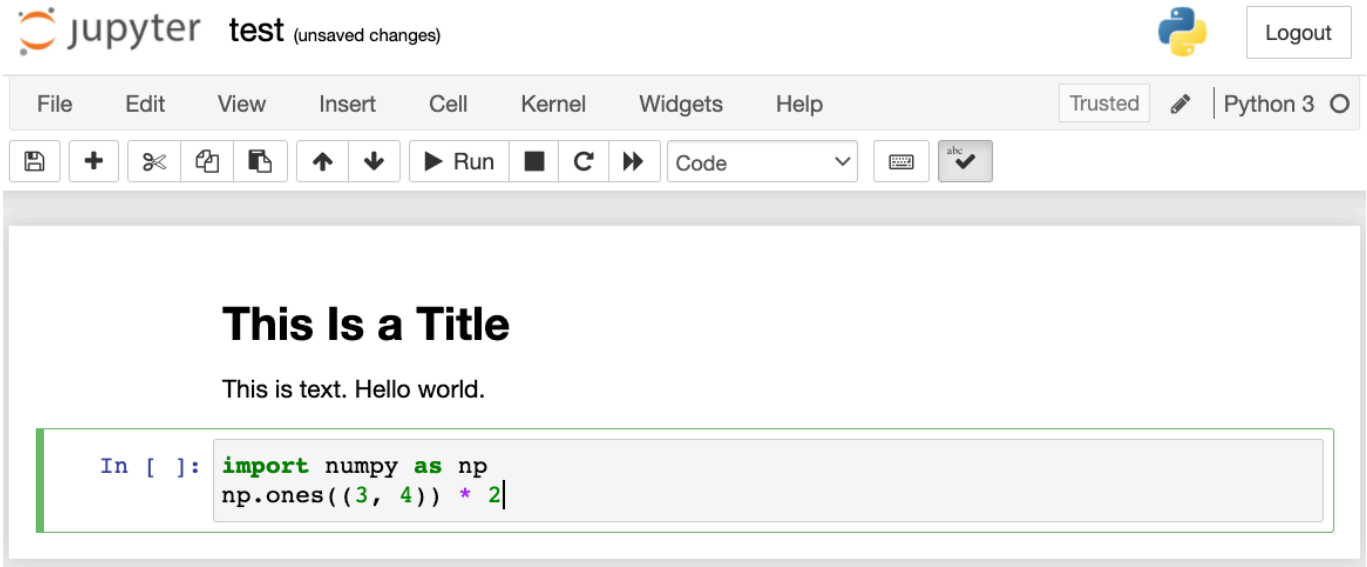
:width: `600px` :label: `fig_jupyter03`

After running, the markdown cell is shown in :numref: `fig_jupyter04` .



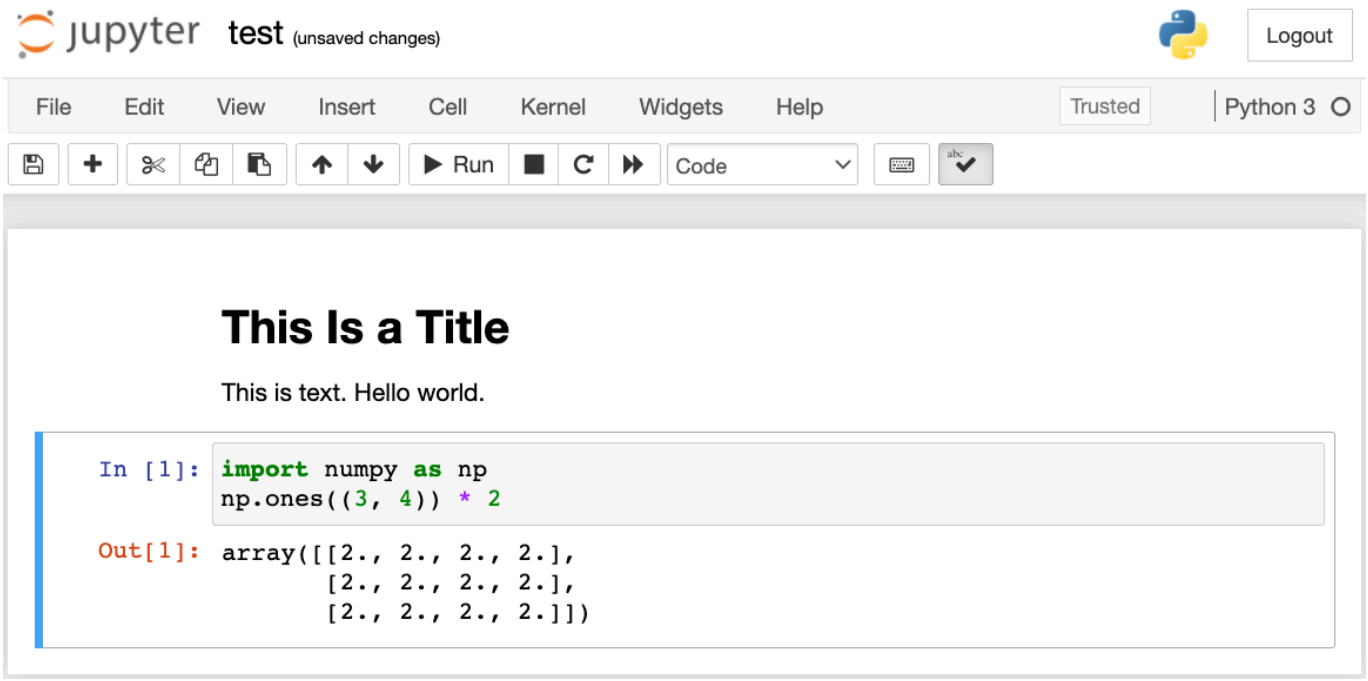
:width: `600px` :label: `fig_jupyter04`

Next, click on the code cell. Multiply the elements by 2 after the last line of code, as shown in :numref: `fig_jupyter05` .



:width: `600px` :label: `fig_jupyter05`

You can also run the cell with a shortcut ("Ctrl + Enter" by default) and obtain the output result from :numref: `fig_jupyter06` .



:width: `600px` :label: `fig_jupyter06`

When a notebook contains more cells, we can click "Kernel" → "Restart & Run All" in the menu bar to run all the cells in the entire notebook. By clicking "Help" → "Edit Keyboard Shortcuts" in the menu bar, you can edit the shortcuts according to your preferences.

Advanced Options

Beyond local editing two things are quite important: editing the notebooks in the markdown format and running Jupyter remotely. The latter matters when we want to run the code on a faster server. The former matters since Jupyter's native ipynb format stores a lot of auxiliary data that is irrelevant to the content, mostly related to how and where the code is run. This is confusing for Git, making reviewing contributions very difficult. Fortunately there is an alternative---native editing in the markdown format.

Markdown Files in Jupyter

If you wish to contribute to the content of this book, you need to modify the source file (md file, not ipynb file) on GitHub. Using the notedown plugin we can modify notebooks in the md format directly in Jupyter.

First, install the notedown plugin, run the Jupyter Notebook, and load the plugin:

```
pip install d2l-notedown  # You may need to uninstall the original notedown.
jupyter notebook --NotebookApp.contents_manager_class='notedown.NotedownContentsManager'
```

You may also turn on the notedown plugin by default whenever you run the Jupyter Notebook. First, generate a Jupyter Notebook configuration file (if it has already been generated, you can skip this step).

```
jupyter notebook --generate-config
```

Then, add the following line to the end of the Jupyter Notebook configuration file (for Linux or macOS, usually in the path `~/.jupyter/jupyter_notebook_config.py`):

```
c.NotebookApp.contents_manager_class = 'notedown.NotedownContentsManager'
```

After that, you only need to run the `jupyter notebook` command to turn on the notedown plugin by default.

Running Jupyter Notebooks on a Remote Server

Sometimes, you may want to run Jupyter notebooks on a remote server and access it through a browser on your local computer. If Linux or macOS is installed on your local machine (Windows can also support this function through third-party software such as PuTTY), you can use port forwarding:

```
ssh myserver -L 8888:localhost:8888
```

The above string `myserver` is the address of the remote server. Then we can use http://localhost:8888 to access the remote server `myserver` that runs Jupyter notebooks. We will detail on how to run Jupyter notebooks on AWS instances later in this appendix.

Timing

We can use the `ExecuteTime` plugin to time the execution of each code cell in Jupyter notebooks. Use the following commands to install the plugin:

```
pip install jupyter_contrib_nbextensions
jupyter contrib nbextension install --user
jupyter nbextension enable execute_time/ExecuteTime
```

Summary

- Using the Jupyter Notebook tool, we can edit, run, and contribute to each section of the book.
- We can run Jupyter notebooks on remote servers using port forwarding.

Exercises

1. Edit and run the code in this book with the Jupyter Notebook on your local machine.
2. Edit and run the code in this book with the Jupyter Notebook *remotely* via port forwarding.
3. Compare the running time of the operations $\mathbf{A}^\top \mathbf{B}$ and $\mathbf{A}\mathbf{B}$ for two square matrices in $\mathbb{R}^{1024 \times 1024}$. Which one is faster?

Discussions

```
import sqlite3



conn = sqlite3.connect(':memory:')  # for temporary DB
# OR to save it:
# conn = sqlite3.connect('my_database.db')
cursor = conn.cursor()



cursor.execute('''
CREATE TABLE sample_sales_data (
    OrderID INTEGER,
    CustomerName TEXT,
    Product TEXT,
    Quantity INTEGER,
    Price REAL,
    OrderDate TEXT
)
''')
conn.commit()



# Step 1: Import necessary libraries
import sqlite3
import pandas as pd

# Step 2: Connect to in-memory SQLite database (or use a file with 'mydata.db')
conn = sqlite3.connect(':memory:')
cursor = conn.cursor()

# Step 3: Create the table
cursor.execute('''
CREATE TABLE sample_sales_data (
    OrderID INTEGER,
    CustomerName TEXT,
    Product TEXT,
    Quantity INTEGER,
    Price REAL,
    OrderDate TEXT
)
''')
conn.commit()
```

```
# Step 4: Insert 20 rows of sample data
sample_data = [
    (1, 'Alice', 'Laptop', 2, 750.00, '2024-01-10'),
    (2, 'Bob', 'Tablet', 1, 300.00, '2024-01-12'),
    (3, 'Charlie', 'Smartphone', 3, 250.00, '2024-01-13'),
    (4, 'David', 'Monitor', 2, 150.00, '2024-01-15'),
    (5, 'Eve', 'Keyboard', 5, 50.00, '2024-01-17'),
    (6, 'Frank', 'Mouse', 4, 25.00, '2024-01-18'),
    (7, 'Grace', 'Laptop', 1, 800.00, '2024-01-20'),
    (8, 'Hannah', 'Tablet', 2, 320.00, '2024-01-21'),
    (9, 'Ivy', 'Monitor', 1, 140.00, '2024-01-22'),
    (10, 'Jack', 'Smartphone', 2, 270.00, '2024-01-23'),
    (11, 'Kara', 'Keyboard', 3, 60.00, '2024-01-25'),
    (12, 'Leo', 'Mouse', 2, 30.00, '2024-01-26'),
    (13, 'Mona', 'Laptop', 1, 850.00, '2024-01-28'),
    (14, 'Nina', 'Tablet', 3, 310.00, '2024-01-29'),
    (15, 'Oscar', 'Smartphone', 1, 260.00, '2024-02-01'),
    (16, 'Paul', 'Monitor', 2, 155.00, '2024-02-02'),
    (17, 'Queen', 'Keyboard', 4, 55.00, '2024-02-04'),
    (18, 'Rick', 'Mouse', 5, 28.00, '2024-02-05'),
    (19, 'Sara', 'Laptop', 1, 900.00, '2024-02-06'),
    (20, 'Tom', 'Tablet', 2, 330.00, '2024-02-07')
]

cursor.executemany("INSERT INTO sample_sales_data VALUES (?, ?, ?, ?, ?, ?)", sample_data
conn.commit()

# Step 5: Display the data using pandas
df = pd.read_sql_query("SELECT * FROM sample_sales_data", conn)
df
```

	OrderID	CustomerName	Product	Quantity	Price	OrderDate
0	1	Alice	Laptop	2	750.0	2024-01-10
1	2	Bob	Tablet	1	300.0	2024-01-12
2	3	Charlie	Smartphone	3	250.0	2024-01-13
3	4	David	Monitor	2	150.0	2024-01-15
4	5	Eve	Keyboard	5	50.0	2024-01-17
5	6	Frank	Mouse	4	25.0	2024-01-18
6	7	Grace	Laptop	1	800.0	2024-01-20
7	8	Hannah	Tablet	2	320.0	2024-01-21
8	9	Ivy	Monitor	1	140.0	2024-01-22
9	10	Jack	Smartphone	2	270.0	2024-01-23
10	11	Kara	Keyboard	3	60.0	2024-01-25
11	12	Leo	Mouse	2	30.0	2024-01-26
12	13	Mona	Laptop	1	850.0	2024-01-28
13	14	Nina	Tablet	3	310.0	2024-01-29
14	15	Oscar	Smartphone	1	260.0	2024-02-01
15	16	Paul	Monitor	2	155.0	2024-02-02
16	17	Queen	Keyboard	4	55.0	2024-02-04
17	18	Rick	Mouse	5	28.0	2024-02-05
18	19	Sara	Laptop	1	900.0	2024-02-06
19	20	Tom	Tablet	2	330.0	2024-02-07

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

```
import pandas as pd
df = pd.read_sql_query("SELECT * FROM sample_sales_data", conn)
df
```

	OrderID	CustomerName	Product	Quantity	Price	OrderDate
0	1	Alice	Laptop	2	750.0	2024-01-10
1	2	Bob	Tablet	1	300.0	2024-01-12
2	3	Charlie	Smartphone	3	250.0	2024-01-13
3	4	David	Monitor	2	150.0	2024-01-15
4	5	Eve	Keyboard	5	50.0	2024-01-17
5	6	Frank	Mouse	4	25.0	2024-01-18
6	7	Grace	Laptop	1	800.0	2024-01-20
7	8	Hannah	Tablet	2	320.0	2024-01-21
8	9	Ivy	Monitor	1	140.0	2024-01-22
9	10	Jack	Smartphone	2	270.0	2024-01-23
10	11	Kara	Keyboard	3	60.0	2024-01-25
11	12	Leo	Mouse	2	30.0	2024-01-26
12	13	Mona	Laptop	1	850.0	2024-01-28
13	14	Nina	Tablet	3	310.0	2024-01-29
14	15	Oscar	Smartphone	1	260.0	2024-02-01
15	16	Paul	Monitor	2	155.0	2024-02-02
16	17	Queen	Keyboard	4	55.0	2024-02-04
17	18	Rick	Mouse	5	28.0	2024-02-05
18	19	Sara	Laptop	1	900.0	2024-02-06
19	20	Tom	Tablet	2	330.0	2024-02-07

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

```
pd.read_sql_query("SELECT CustomerName, Product, Price FROM sample_sales_data", conn)
```

	CustomerName	Product	Price
0	Alice	Laptop	750.0
1	Bob	Tablet	300.0
2	Charlie	Smartphone	250.0
3	David	Monitor	150.0
4	Eve	Keyboard	50.0
5	Frank	Mouse	25.0
6	Grace	Laptop	800.0
7	Hannah	Tablet	320.0
8	Ivy	Monitor	140.0
9	Jack	Smartphone	270.0
10	Kara	Keyboard	60.0
11	Leo	Mouse	30.0
12	Mona	Laptop	850.0

```
pd.read_sql_query("SELECT * FROM sample_sales_data WHERE Price > 20000", conn)
```

	OrderID	CustomerName	Product	Quantity	Price	OrderDate

```
pd.read_sql_query("SELECT SUM(Quantity * Price) AS Total_Revenue FROM sample_sales_data",
```

	Total_Revenue
0	9830.0

```
pd.read_sql_query("SELECT Product, AVG(Price) AS Avg_Price FROM sample_sales_data GROUP B
```

	Product	Avg_Price
0	Keyboard	55.000000
1	Laptop	825.000000
2	Monitor	148.333333
3	Mouse	27.666667