



## **Electronic Assignment Cover Sheet**

### **Student(s) name as per the student card:**

Harshita Sunil Tewani: 10609984

Vaishnavi Janardhan Chavan: 10612841

Sushil Babulal Jangid: 10615112

Course Title: Data Analytics

Lecturer Name: Amit Sharma

Module/Subject Title: Programming for Data Analysis

Assignment Title: CA2

### **✓ Introduction**

To design and develop a Data Acquisition and Pre-processing Pipeline, we used Flipkart India's website and performed web scraping on it. For this first we had to import necessary libraries such as BeautifulSoup for web scraping, Pandas for data manipulation, and Requests for HTTP requests.

```
# Import the pandas library as pd
import pandas as pd

# Import the requests library for making HTTP requests
import requests

# Import the BeautifulSoup library for web scraping
import bs4
```

## ✓ Data Acquisition

Next, we'll specify the URL of the Flipkart website to be scraped, followed by a GET request to retrieve the HTML content.

Here, we keep the page number as an empty space so that we can loop over it and scrap the data of multiple pages.

```
url = "https://www.flipkart.com/search?q=iphone+&otracker=search&otracker1=search"
res = requests.get(url)
res
```

➡ <Response [200]>

The above output of res code '200' typically means that the request was successful and the server returned the expected response.

The HTML content will then be parsed and the relevant data extracted using BeautifulSoup. In this scenario, we'll extract each iPhone's name, original price, ratings, offer percentage and offer price.

```
soup = bs4.BeautifulSoup(res.text, 'html.parser')
```

The script begins by creating an empty list called 'Mobile\_Details'. It then loops through the website's pages, with **range(1, 19)** specifying that pages 1 through 18 will be scraped.

For each page, the script uses the requests library to send a request to the website and obtains the HTML content from the response object's text attribute. After that, the BeautifulSoup function from the BeautifulSoup library is used to parse the HTML text and generate a BeautifulSoup object bs.

The script then looks for all div elements with the class "\_2kHMtA" using the BeautifulSoup object's 'findAll()' method.

Then the script extracts the 'Product\_Name', 'Original\_Price\_₹', 'Product\_Color', 'Storage', 'Display', 'Camera', 'Processor', 'Offer\_Price\_₹', 'Offer\_%' and 'Ratings' from the website and appends all the

data to the empty dictionary 'record' that we create at each iteration.

This dictionary, record, will in turn be appended to the 'Mobile\_Details' list before moving onto the next iteration.

```
Mobile_Details = []

for pages in range(1, 19):
    html = requests.get(url + str(pages))
    bs = bs4.BeautifulSoup(html.text)
    for i in bs.findAll("div", {"class" : "_2kHMtA"}):
        if i.find('div', class_ = '_2tfzpE'):
            break

    record = {}
    record['Product_Name'] = i.find('div', class_ = '_4rR01T').text.split(' (', 1)
    record['Product_Color'] = i.find('div', class_ = '_4rR01T').text.split(' (', 1)

    li_tags = bs.find_all('li')
    record['Storage'] = li_tags[0].text.strip()
    record['Display'] = li_tags[1].text.strip()
    record['Camera'] = li_tags[2].text.strip()
    record['Processor'] = li_tags[3].text.strip()

    # Checking if the product has offer or no
    if i.find('div', class_ = '_3I9_wc _27UcVY'):
        record['Original_Price_₹'] = i.find('div', class_ = '_3I9_wc _27UcVY').text
        record['Offer_Price_₹'] = i.find('div', class_ = '_30jeq3 _1_WHN1').text.strip()
        record['Offer_%'] = i.find('div', class_ = '_3Ay6Sb').text.strip()
    else:
        record['Original_Price_₹'] = i.find('div', class_ = '_30jeq3 _1_WHN1').text
        # At this point when there's no offer, we copy the original price to offer
        record['Offer_Price_₹'] = record['Original_Price_₹']

    record['Ratings'] = i.find('div', class_ = '_3LWZlK').text.strip()

    Mobile_Details.append(record)
```

The 'pd.json\_normalize()' function takes this list and creates a table where each dictionary key becomes a column and its corresponding value becomes the cell value.

```
iPhone_DF = pd.json_normalize(Mobile_Details)
```

To further confirm, we have a quick look at this dataframe.

```
iPhone_DF
```



	Product_Name	Product_Color	Storage	Display	Camera	Processor	Original
0	APPLE iPhone 13	Green	128 GB ROM	15.49 cm (6.1 inch) Super Retina XDR Display	12MP + 12MP   12MP Front Camera	A15 Bionic Chip Processor	
1	APPLE iPhone 13	Pink	128 GB ROM	15.49 cm (6.1 inch) Super Retina XDR Display	12MP + 12MP   12MP Front Camera	A15 Bionic Chip Processor	
2	APPLE iPhone 13	Midnight	128 GB ROM	15.49 cm (6.1 inch) Super Retina XDR Display	12MP + 12MP   12MP Front Camera	A15 Bionic Chip Processor	
3	APPLE iPhone 13	Blue	128 GB ROM	15.49 cm (6.1 inch) Super Retina XDR Display	12MP + 12MP   12MP Front Camera	A15 Bionic Chip Processor	
4	APPLE iPhone 13	Starlight	128 GB ROM	15.49 cm (6.1 inch) Super Retina XDR Display	12MP + 12MP   12MP Front Camera	A15 Bionic Chip Processor	

The output of 'iPhone\_DF.info()' includes the number of rows and columns in the DataFrame, the name of each column, the number of non-null values in each column, and the data type of each column.

iPhone\_DF.info()




```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 428 entries, 0 to 427
Data columns (total 10 columns):
#   Column                Non-Null Count  Dtype
---  -
0   Product_Name          428 non-null    object
1   Product_Color          428 non-null    object
2   Storage                428 non-null    object
3   Display                428 non-null    object
4   Camera                 428 non-null    object
5   Processor              428 non-null    object
6   Original_Price_₹       428 non-null    object
7   Offer_Price_₹          428 non-null    object
8   Offer_%                292 non-null    object
9   Ratings                428 non-null    object
dtypes: object(10)
```

memory usage: 33.6+ KB

'iPhone\_DF.describe()' is used to get an overview of the count of data, frequency and uniqueness of data in the iPhone\_DF DataFrame.

```
iPhone_DF.describe()
```



	Product_Name	Product_Color	Storage	Display	Camera	Processor	Original_Price_₹
count	428	428	428	428	428	428	428
unique	34	29	4	9	5	8	428
top	APPLE iPhone 13 Pro Max	Silver	256 GB ROM	15.49 cm (6.1 inch) Super Retina XDR Display	12MP + 12MP   12MP Front Camera	A14 Bionic Chip with Next Generation Neural En...	428

## ✓ Data Preprocessing and Transformation

After going through the description of the iPhone\_DF dataframe, we now preprocess the columns. This includes removing the ("₹", ",", and "% off") from the columns using the 'str.replace' method

```
iPhone_DF['Original_Price_₹'] = iPhone_DF['Original_Price_₹'].str.replace('₹', '')
iPhone_DF['Offer_Price_₹'] = iPhone_DF['Offer_Price_₹'].str.replace('₹', '')
```


```
iPhone_DF['Original_Price_₹'] = iPhone_DF['Original_Price_₹'].str.replace(',', '')
iPhone_DF['Offer_Price_₹'] = iPhone_DF['Offer_Price_₹'].str.replace(',', '')
```

```
iPhone_DF['Offer_%'] = iPhone_DF['Offer_%'].str.replace('% off', '')
```

Furthermore, we converted the datatype of 'Original\_Price\_₹', 'Offer\_Price\_₹', 'Ratings', and 'Offer\_%' from object to float.

```
iPhone_DF['Original_Price_₹'] = iPhone_DF['Original_Price_₹'].astype('float')
iPhone_DF['Offer_Price_₹'] = iPhone_DF['Offer_Price_₹'].astype('float')
iPhone_DF['Ratings'] = iPhone_DF['Ratings'].astype('float')
iPhone_DF['Offer_%'] = iPhone_DF['Offer_%'].astype('float')
```

```
iPhone_DF.info()
```



```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 428 entries, 0 to 427
Data columns (total 10 columns):
```

#	Column	Non-Null Count	Dtype
0	Product_Name	428 non-null	object
1	Product_Color	428 non-null	object
2	Storage	428 non-null	object
3	Display	428 non-null	object
4	Camera	428 non-null	object
5	Processor	428 non-null	object
6	Original_Price_₹	428 non-null	float64
7	Offer_Price_₹	428 non-null	float64
8	Offer_%	292 non-null	float64
9	Ratings	428 non-null	float64

dtypes: float64(4), object(6)  
memory usage: 33.6+ KB

Next, we check if there are any null values in the dataframe.

```
iPhone_DF.isna().sum().sum()
```

⇒ 136

If any null values are found, we replace them with 0.

As we have already seen the dataset and we know only the Offer\_% column has missing values as some phones do not have offers and we have copied the original price to the offer price at the time of data retrieval, now we make sure we fill in the Offer\_% to 0.

```
iPhone_DF['Offer_%'].isna().sum().sum()
```

⇒ 136

```
iPhone_DF['Offer_%'].fillna(0, inplace=True)
```

As we can see, there are no null values now.

```
iPhone_DF.isna().sum().sum()
```

⇒ 0

```
iPhone_DF
```



	Product_Name	Product_Color	Storage	Display	Camera	Processor	Original
0	APPLE iPhone 13	Green	128 GB ROM	15.49 cm (6.1 inch) Super Retina XDR Display	12MP + 12MP   12MP Front Camera	A15 Bionic Chip Processor	
1	APPLE iPhone 13	Pink	128 GB ROM	15.49 cm (6.1 inch) Super Retina XDR Display	12MP + 12MP   12MP Front Camera	A15 Bionic Chip Processor	
2	APPLE iPhone 13	Midnight	128 GB ROM	15.49 cm (6.1 inch) Super Retina XDR Display	12MP + 12MP   12MP Front Camera	A15 Bionic Chip Processor	
3	APPLE iPhone 13	Blue	128 GB ROM	15.49 cm (6.1 inch) Super Retina XDR Display	12MP + 12MP   12MP Front Camera	A15 Bionic Chip Processor	
4	APPLE iPhone 13	Starlight	128 GB ROM	15.49 cm (6.1 inch) Super Retina XDR Display	12MP + 12MP   12MP Front Camera	A15 Bionic Chip Processor	

## ✓ Querying Data using DataFrame

**Q.** Finding a White color iPhone within ₹60000.

```
condition_1 = (iPhone_DF['Original_Price_₹'] < 60000) & \
               (iPhone_DF['Product_Color'] == 'White')
iPhone_DF[condition_1]
```



	Product_Name	Product_Color	Storage	Display	Camera	Processor	Original
6	APPLE iPhone 11	White	128 GB ROM	15.49 cm (6.1 inch) Super Retina XDR Display	12MP + 12MP   12MP Front Camera	A15 Bionic Chip Processor	
13	APPLE iPhone 11	White	128 GB ROM	15.49 cm (6.1 inch) Super Retina XDR Display	12MP + 12MP   12MP Front Camera	A15 Bionic Chip Processor	
34	APPLE iPhone 12	White	64 GB ROM	15.49 cm (6.1 inch) Liquid Retina HD Display	12MP + 12MP   12MP Front Camera	A13 Bionic Chip Processor	
127	APPLE iPhone XR	White	256 GB ROM	17.02 cm (6.7 inch) Super Retina XDR Display	48MP + 12MP + 12MP   12MP Front Camera	A16 Bionic Chip, 6 Core Processor	
139	APPLE iPhone 12 mini	White	256 GB ROM	17.02 cm (6.7 inch) Super Retina XDR Display	48MP + 12MP + 12MP   12MP Front Camera	A16 Bionic Chip, 6 Core Processor	
167	APPLE iPhone SE	White	64 GB ROM	13.97 cm (5.5 inch) Retina HD Display	12MP + 12MP   7MP Front Camera	A11 Bionic Chip with 64-bit Architecture, Neur...	
182	Apple iPhone XR	White	64 GB ROM	13.97 cm (5.5 inch) Retina HD Display	12MP + 12MP   7MP Front Camera	A11 Bionic Chip with 64-bit Architecture, Neur...	
191	Apple iPhone SE	White	64 GB ROM	13.97 cm (5.5 inch) Retina HD Display	12MP + 12MP   7MP Front Camera	A11 Bionic Chip with 64-bit Architecture, Neur...	
233	APPLE iPhone SE	White	512 GB ROM	14.73 cm (5.8 inch) Super Retina XDR Display	12MP + 12MP + 12MP   12MP Front Camera	A13 Bionic Chip Processor	



245	APPLE iPhone SE	White	256 GB ROM	15.49 cm (6.1 inch) Super Retina XDR Display	48MP + 12MP + 12MP 12MP Front Camera	A16 Bionic Chip, 6 Core Processor
-----	-----------------	-------	------------	----------------------------------------------	--------------------------------------	-----------------------------------

**Q.** Finding an iPhone on a discount of more than 17% and with price more than ₹90000

```
condition_2 = (iPhone_DF['Offer_Price_₹'] > 90000) & \
              (iPhone_DF['Offer_%'] >= 17)
iPhone_DF[condition_2]
```



	Product_Name	Product_Color	Storage	Display	Camera	Processor	Original
108	APPLE iPhone 12 Pro	Gold	128 GB ROM	11.94 cm (4.7 inch) Retina HD Display	12MP Rear Camera   7MP Front Camera	A13 Bionic Chip with 3rd Gen Neural Engine Pro...	
122	APPLE iPhone 11 Pro Max	Gold	256 GB ROM	17.02 cm (6.7 inch) Super Retina XDR Display	48MP + 12MP + 12MP   12MP Front Camera	A16 Bionic Chip, 6 Core Processor Processor	
131	APPLE iPhone 11 Pro Max	Space Grey	256 GB ROM	17.02 cm (6.7 inch) Super Retina XDR Display	48MP + 12MP + 12MP   12MP Front Camera	A16 Bionic Chip, 6 Core Processor Processor	
171	APPLE iPhone 12 Pro	Silver	64 GB ROM	13.97 cm (5.5 inch) Retina HD Display	12MP + 12MP   7MP Front Camera	A11 Bionic Chip with 64-bit Architecture, Neur...	
183	APPLE iPhone 12 Pro	Graphite	64 GB ROM	13.97 cm (5.5 inch) Retina HD Display	12MP + 12MP   7MP Front Camera	A11 Bionic Chip with 64-bit Architecture, Neur...	
353	APPLE iPhone 11 Pro Max	Midnight Green	256 GB ROM	11.94 cm (4.7 inch) Retina HD Display	12MP Rear Camera   7MP Front Camera	A10 Fusion Chip with 64-bit Architecture and E...	
361	APPLE iPhone 11 Pro Max	Midnight Green	64 GB ROM	15.49 cm (6.1 inch) Display	12MP Rear Camera   7MP Front Camera	A12 Bionic Chip Processor	
371	APPLE iPhone 11 Pro Max	Silver	64 GB ROM	15.49 cm (6.1 inch) Display	12MP Rear Camera   7MP Front Camera	A12 Bionic Chip Processor	
380	APPLE iPhone 12 Pro Max	Graphite	64 GB ROM	15.49 cm (6.1 inch) Display	12MP Rear Camera   7MP Front	A12 Bionic Chip Processor	

381	APPLE iPhone 12 Pro Max	Silver	64 GB ROM	15.49 cm (6.1 inch) Display	Camera	A12 Bionic Chip Processor
					12MP Rear	
					Camera 17MP Front	

**Q.** Finding an iPhone with rating higher than 4.5, on more than 15% discount and within ₹70000.

```
condition_3 = (iPhone_DF['Ratings'] > 4.5) & \
               ((iPhone_DF['Offer_Price_₹'] < 70000) & \
                (iPhone_DF['Offer_%'] > 15))
iPhone_DF[condition_3]
```



	Product_Name	Product_Color	Storage	Display	Camera	Processor	Original
88	Apple iPhone XR	(PRODUCT)RED	64 GB ROM	15.49 cm (6.1 inch) Super Retina XDR Display	12MP + 12MP   12MP Front Camera	A14 Bionic Chip with Next Generation Neural En...	
109	Apple iPhone XR	(PRODUCT)RED	128 GB ROM	11.94 cm (4.7 inch) Retina HD Display	12MP Rear Camera   7MP Front Camera	A13 Bionic Chip with 3rd Gen Neural Engine Pro...	
121	Apple iPhone XR	Blue	256 GB ROM	17.02 cm (6.7 inch) Super Retina XDR Display	48MP + 12MP + 12MP   12MP Front Camera	A16 Bionic Chip, 6 Core Processor Processor	
127	APPLE iPhone XR	White	256 GB ROM	17.02 cm (6.7 inch) Super Retina XDR Display	48MP + 12MP + 12MP   12MP Front Camera	A16 Bionic Chip, 6 Core Processor Processor	
129	APPLE iPhone 8 Plus	Space Grey	256 GB ROM	17.02 cm (6.7 inch) Super Retina XDR Display	48MP + 12MP + 12MP   12MP Front Camera	A16 Bionic Chip, 6 Core Processor Processor	
134	APPLE iPhone XR	(PRODUCT)RED	256 GB ROM	17.02 cm (6.7 inch) Super Retina XDR Display	48MP + 12MP + 12MP   12MP Front Camera	A16 Bionic Chip, 6 Core Processor Processor	
144	APPLE iPhone 8 Plus	Gold	64 GB ROM	13.97 cm (5.5 inch) Retina HD Display	12MP + 12MP   7MP Front Camera	A11 Bionic Chip with 64-bit Architecture, Neur...	
168	APPLE iPhone 8 Plus	Silver	64 GB ROM	13.97 cm (5.5 inch) Retina HD Display	12MP + 12MP   7MP Front Camera	A11 Bionic Chip with 64-bit Architecture, Neur...	
181	Apple iPhone XR	Black	64 GB ROM	13.97 cm (5.5 inch) Retina HD Display	12MP + 12MP   7MP Front Camera	A11 Bionic Chip with 64-bit Architecture, Neur...	

182	Apple iPhone XR	White	64 GB ROM	13.97 cm (5.5 inch) Retina HD Display	12MP + 12MP   7MP Front Camera	A11 Bionic Chip with 64-bit Architecture, Neur...
212	Apple iPhone XR	Yellow	256 GB ROM	13.72 cm (5.4 inch) Super Retina XDR Display	12MP + 12MP   12MP Front Camera	A15 Bionic Chip Processor
				15.49 cm (6.1 inch)	48MP + 12MP +	A16 Bionic Chip

**Q.** Finding an iPhone with 128GB storage, color - blue or midnight and within ₹80000.

```
condition_4 = (iPhone_DF['Storage'] == '128 GB ROM') & \
              ((iPhone_DF['Product_Color'] == 'Blue') | \
               (iPhone_DF['Product_Color'] == 'Midnight')) & \
              (iPhone_DF['Offer_Price_₹'] < 80000)
iPhone_DF[condition_4]
```



	Product_Name	Product_Color	Storage	Display	Camera	Processor	Original_
2	APPLE iPhone 13	Midnight	128 GB ROM	15.49 cm (6.1 inch) Super Retina XDR Display	12MP + 12MP   12MP Front Camera	A15 Bionic Chip Processor	
3	APPLE iPhone 13	Blue	128 GB ROM	15.49 cm (6.1 inch) Super Retina XDR Display	12MP + 12MP   12MP Front Camera	A15 Bionic Chip Processor	
7	APPLE iPhone 14	Midnight	128 GB ROM	15.49 cm (6.1 inch) Super Retina XDR Display	12MP + 12MP   12MP Front Camera	A15 Bionic Chip	

## ✓ Converting the DataFrame to a .csv file

Using the '.to\_csv' method to convert the dataframe to a csv file.

```
iPhone_DF.to_csv('iphones.csv', index = False)
```

```
iPhone_csv = pd.read_csv('iphones.csv')
```

Reading the CSV file.

iPhone\_csv



	Product_Name	Product_Color	Storage	Display	Camera	Processor	Original
0	APPLE iPhone 13	Green	128 GB ROM	15.49 cm (6.1 inch) Super Retina XDR Display	12MP + 12MP   12MP Front Camera	A15 Bionic Chip Processor	
1	APPLE iPhone 13	Pink	128 GB ROM	15.49 cm (6.1 inch) Super Retina XDR Display	12MP + 12MP   12MP Front Camera	A15 Bionic Chip Processor	
2	APPLE iPhone 13	Midnight	128 GB ROM	15.49 cm (6.1 inch) Super Retina XDR Display	12MP + 12MP   12MP Front Camera	A15 Bionic Chip Processor	
3	APPLE iPhone 13	Blue	128 GB ROM	15.49 cm (6.1 inch) Super Retina XDR Display	12MP + 12MP   12MP Front Camera	A15 Bionic Chip Processor	
4	APPLE iPhone 13	Starlight	128 GB ROM	15.49 cm (6.1 inch) Super Retina XDR Display	12MP + 12MP   12MP Front Camera	A15 Bionic Chip Processor	

## ✓ Loading the data into Database

Below code imports the sqlite3 library, which provides a way to connect to and interact with SQLite databases using Python.

The 'conn' creates a new connection to a database named iphone.db. If the database does not already exist, it will be created automatically. If the file already exists, it will be opened for reading and writing.

The cursor object, which allows us to execute SQL commands on the database. We can use the cursor to execute queries and fetch results from the database.

```
import sqlite3
```

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

Before we create our 'iPhoneData' table we drop the table if it exists, in case of multiple runs of the file.

Now, this will let us create the table again without any errors/interruptions for the further steps.

```
c.execute('DROP TABLE IF EXISTS iPhoneData')
```

```
↳ <sqlite3.Cursor at 0x7f8dcaebbb90>
```

Using the '.execute()' we create a new table named 'iPhoneData' in the connected SQLite database, with the specified column names and data types:

```
c.execute('''CREATE TABLE iPhoneData (  
            product_name text,  
            product_color text,  
            storage text,  
            display text,  
            camera text,  
            processor text,  
            original_price real,  
            offer_price real,  
            offer_perc real,  
            ratings real  
        )''')
```

```
↳ <sqlite3.Cursor at 0x7f8dcaebbb90>
```

The code is iterating through each row of the iPhone\_DF DataFrame using the iter tuples() method and to insert the values into the table we use 'c.execute' statement.

```
for row in iPhone_DF.iteruples(index=False):  
    c.execute('INSERT INTO iPhoneData VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?)', row)
```

For printing all the rows, we use for loop of 'SELECT \* FROM' statement in 'c.execute'.

```
for row in c.execute('SELECT * FROM iPhoneData'):  
    print(row)
```

```
↳ ('APPLE iPhone 13', 'Green', '128 GB ROM', '15.49 cm (6.1 inch) Super Retina )  
('APPLE iPhone 13', 'Pink', '128 GB ROM', '15.49 cm (6.1 inch) Super Retina XI  
('APPLE iPhone 13', 'Midnight', '128 GB ROM', '15.49 cm (6.1 inch) Super Reti  
('APPLE iPhone 13', 'Blue', '128 GB ROM', '15.49 cm (6.1 inch) Super Retina XI  
('APPLE iPhone 13', 'Starlight', '128 GB ROM', '15.49 cm (6.1 inch) Super Ret:
```

```

('APPLE iPhone 11', 'Black', '128 GB ROM', '15.49 cm (6.1 inch) Super Retina )
('APPLE iPhone 11', 'White', '128 GB ROM', '15.49 cm (6.1 inch) Super Retina )
('APPLE iPhone 14', 'Midnight', '128 GB ROM', '15.49 cm (6.1 inch) Super Reti
('APPLE iPhone 14', 'Blue', '128 GB ROM', '15.49 cm (6.1 inch) Super Retina XI
('APPLE iPhone 14', 'Midnight', '128 GB ROM', '15.49 cm (6.1 inch) Super Reti
('APPLE iPhone 14', '(PRODUCT)RED', '128 GB ROM', '15.49 cm (6.1 inch) Super I
('APPLE iPhone 14 Plus', 'Purple', '128 GB ROM', '15.49 cm (6.1 inch) Super R
('APPLE iPhone 14 Plus', 'Blue', '128 GB ROM', '15.49 cm (6.1 inch) Super Reti
('APPLE iPhone 11', 'White', '128 GB ROM', '15.49 cm (6.1 inch) Super Retina )
('APPLE iPhone 14 Plus', 'Starlight', '128 GB ROM', '15.49 cm (6.1 inch) Supe
('APPLE iPhone 14', 'Purple', '128 GB ROM', '15.49 cm (6.1 inch) Super Retina
('APPLE iPhone 14 Plus', '(PRODUCT)RED', '128 GB ROM', '15.49 cm (6.1 inch) S
('APPLE iPhone 14', 'Blue', '128 GB ROM', '15.49 cm (6.1 inch) Super Retina XI
('APPLE iPhone 13', '(PRODUCT)RED', '128 GB ROM', '15.49 cm (6.1 inch) Super I
('APPLE iPhone 14 Pro', 'Space Black', '128 GB ROM', '15.49 cm (6.1 inch) Supe
('APPLE iPhone 11', 'Black', '128 GB ROM', '15.49 cm (6.1 inch) Super Retina )
('APPLE iPhone 14', 'Starlight', '128 GB ROM', '15.49 cm (6.1 inch) Super Reti
('APPLE iPhone 13', 'Midnight', '128 GB ROM', '15.49 cm (6.1 inch) Super Reti
('APPLE iPhone 14', 'Yellow', '128 GB ROM', '15.49 cm (6.1 inch) Super Retina
('APPLE iPhone 11', 'Black', '64 GB ROM', '15.49 cm (6.1 inch) Liquid Retina I
('APPLE iPhone 13', 'Blue', '64 GB ROM', '15.49 cm (6.1 inch) Liquid Retina HI
('APPLE iPhone 14 Pro Max', 'Space Black', '64 GB ROM', '15.49 cm (6.1 inch) I
('APPLE iPhone 14', 'Starlight', '64 GB ROM', '15.49 cm (6.1 inch) Liquid Reti
('APPLE iPhone 13', '(PRODUCT)RED', '64 GB ROM', '15.49 cm (6.1 inch) Liquid I
('APPLE iPhone 12', 'Blue', '64 GB ROM', '15.49 cm (6.1 inch) Liquid Retina HI
('APPLE iPhone 14 Plus', 'Midnight', '64 GB ROM', '15.49 cm (6.1 inch) Liquid
('APPLE iPhone 14 Plus', '(PRODUCT)RED', '64 GB ROM', '15.49 cm (6.1 inch) Liq
('APPLE iPhone 14 Pro', 'Gold', '64 GB ROM', '15.49 cm (6.1 inch) Liquid Reti
('APPLE iPhone 14', 'Purple', '64 GB ROM', '15.49 cm (6.1 inch) Liquid Retina
('APPLE iPhone 12', 'White', '64 GB ROM', '15.49 cm (6.1 inch) Liquid Retina I
('APPLE iPhone 14', '(PRODUCT)RED', '64 GB ROM', '15.49 cm (6.1 inch) Liquid I
('APPLE iPhone 12', 'Black', '64 GB ROM', '15.49 cm (6.1 inch) Liquid Retina I
('APPLE iPhone 14 Pro Max', 'Gold', '64 GB ROM', '15.49 cm (6.1 inch) Liquid I
('APPLE iPhone 13', '(PRODUCT)RED', '64 GB ROM', '15.49 cm (6.1 inch) Liquid I
('APPLE iPhone 14 Pro Max', 'Silver', '64 GB ROM', '15.49 cm (6.1 inch) Liqui
('APPLE iPhone 13 mini', 'Blue', '64 GB ROM', '15.49 cm (6.1 inch) Liquid Reti
('APPLE iPhone 14 Pro', 'Silver', '64 GB ROM', '15.49 cm (6.1 inch) Liquid Re
('APPLE iPhone 14 Pro', 'Deep Purple', '64 GB ROM', '15.49 cm (6.1 inch) Liqu
('APPLE iPhone 14', 'Starlight', '64 GB ROM', '15.49 cm (6.1 inch) Liquid Reti
('APPLE iPhone 12', 'Red', '64 GB ROM', '15.49 cm (6.1 inch) Liquid Retina HD
('APPLE iPhone 11', 'Red', '64 GB ROM', '15.49 cm (6.1 inch) Liquid Retina HD
('APPLE iPhone 11', 'Red', '64 GB ROM', '15.49 cm (6.1 inch) Liquid Retina HD
('APPLE iPhone 11', 'Green', '64 GB ROM', '15.49 cm (6.1 inch) Liquid Retina I
('APPLE iPhone 12', 'Green', '256 GB ROM', '15.49 cm (6.1 inch) Super Retina )
('APPLE iPhone 14', 'Midnight', '256 GB ROM', '15.49 cm (6.1 inch) Super Reti
('APPLE iPhone 14 Plus', 'Starlight', '256 GB ROM', '15.49 cm (6.1 inch) Supe
('APPLE iPhone 12', 'Black', '256 GB ROM', '15.49 cm (6.1 inch) Super Retina )
('APPLE iPhone 14 Pro', 'Deep Purple', '256 GB ROM', '15.49 cm (6.1 inch) Supe
('APPLE iPhone 13 Pro', 'Gold', '256 GB ROM', '15.49 cm (6.1 inch) Super Reti
('APPLE iPhone 14 Plus', 'Purple', '256 GB ROM', '15.49 cm (6.1 inch) Super R
('APPLE iPhone 13 Pro', 'Silver', '256 GB ROM', '15.49 cm (6.1 inch) Super Re
('APPLE iPhone 13', 'Blue', '256 GB ROM', '15.49 cm (6.1 inch) Super Retina XI
('APPLE iPhone 11', 'Purple', '256 GB ROM', '15.49 cm (6.1 inch) Super Retina

```

**Q.** SQL statement to sort the "ratings" column in descending order



```
query_1 = '''SELECT product_name, product_color, original_price, offer_price, rat
              ORDER BY ratings DESC'''
```

```
for row in c.execute(query_1):
    print(row)
```

```
➡ ('APPLE iPhone 7 Plus', 'PRODUCT) (Red', 92000.0, 85400.0, 5.0)
('APPLE iPhone 13', 'Green', 69900.0, 61999.0, 4.7)
('APPLE iPhone 13', 'Pink', 69900.0, 61999.0, 4.7)
('APPLE iPhone 13', 'Midnight', 69900.0, 61999.0, 4.7)
('APPLE iPhone 13', 'Blue', 69900.0, 61999.0, 4.7)
('APPLE iPhone 13', 'Starlight', 69900.0, 61999.0, 4.7)
('APPLE iPhone 14 Plus', 'Purple', 89900.0, 79999.0, 4.7)
('APPLE iPhone 14 Plus', 'Blue', 89900.0, 79999.0, 4.7)
('APPLE iPhone 14 Plus', 'Starlight', 89900.0, 79999.0, 4.7)
('APPLE iPhone 14 Plus', '(PRODUCT)RED', 89900.0, 77999.0, 4.7)
('APPLE iPhone 13', '(PRODUCT)RED', 69900.0, 61999.0, 4.7)
('APPLE iPhone 14 Pro', 'Space Black', 129900.0, 119999.0, 4.7)
('APPLE iPhone 13', 'Midnight', 79900.0, 71999.0, 4.7)
('APPLE iPhone 13', 'Blue', 79900.0, 71999.0, 4.7)
('APPLE iPhone 14 Pro Max', 'Space Black', 139900.0, 127999.0, 4.7)
('APPLE iPhone 13', '(PRODUCT)RED', 69900.0, 61999.0, 4.7)
('APPLE iPhone 14 Plus', 'Midnight', 89900.0, 79999.0, 4.7)
('APPLE iPhone 14 Plus', '(PRODUCT)RED', 99900.0, 89999.0, 4.7)
('APPLE iPhone 14 Pro', 'Gold', 129900.0, 119999.0, 4.7)
('APPLE iPhone 14 Pro Max', 'Gold', 139900.0, 127999.0, 4.7)
('APPLE iPhone 13', '(PRODUCT)RED', 79900.0, 71999.0, 4.7)
('APPLE iPhone 14 Pro Max', 'Silver', 139900.0, 127999.0, 4.7)
('APPLE iPhone 14 Pro', 'Silver', 129900.0, 119999.0, 4.7)
('APPLE iPhone 14 Pro', 'Deep Purple', 129900.0, 119999.0, 4.7)
('APPLE iPhone 14 Plus', 'Starlight', 119900.0, 109999.0, 4.7)
('APPLE iPhone 14 Pro', 'Deep Purple', 159900.0, 149999.0, 4.7)
('APPLE iPhone 13 Pro', 'Gold', 149900.0, 125999.0, 4.7)
('APPLE iPhone 14 Plus', 'Purple', 119900.0, 109999.0, 4.7)
('APPLE iPhone 13 Pro', 'Silver', 149900.0, 125999.0, 4.7)
('APPLE iPhone 13', 'Blue', 99900.0, 91999.0, 4.7)
('APPLE iPhone 14 Plus', 'Midnight', 99900.0, 89999.0, 4.7)
('APPLE iPhone 14 Pro', 'Space Black', 179900.0, 169999.0, 4.7)
('APPLE iPhone 14 Plus', 'Blue', 119900.0, 109999.0, 4.7)
('APPLE iPhone 14 Plus', '(PRODUCT)RED', 119900.0, 109999.0, 4.7)
('APPLE iPhone 13', 'Green', 99900.0, 91999.0, 4.7)
('APPLE iPhone 14 Pro Max', 'Silver', 169900.0, 157999.0, 4.7)
('APPLE iPhone 14 Pro', 'Space Black', 179900.0, 169999.0, 4.7)
('APPLE iPhone 14 Pro', 'Space Black', 159900.0, 149999.0, 4.7)
('APPLE iPhone 14 Pro Max', 'Gold', 169900.0, 157999.0, 4.7)
('APPLE iPhone 13', 'Starlight', 99900.0, 91999.0, 4.7)
('APPLE iPhone 14 Plus', 'Starlight', 99900.0, 89999.0, 4.7)
('APPLE iPhone 14 Plus', 'Yellow', 89900.0, 79999.0, 4.7)
('APPLE iPhone 13 Pro', 'Silver', 169900.0, 144999.0, 4.7)
('APPLE iPhone 13', 'Starlight', 99900.0, 91999.0, 4.7)
('APPLE iPhone 14 Pro Max', 'Gold', 169900.0, 157999.0, 4.7)
('APPLE iPhone 14 Pro Max', 'Space Black', 149900.0, 137999.0, 4.7)
('APPLE iPhone 13', 'Pink', 99900.0, 91999.0, 4.7)
('APPLE iPhone 14 Plus', 'Purple', 99900.0, 89999.0, 4.7)
('APPLE iPhone 14 Plus', 'Starlight', 99900.0, 89999.0, 4.7)
('APPLE iPhone 14 Plus', 'Blue', 99900.0, 89999.0, 4.7)
('APPLE iPhone 14 Pro', 'Gold', 179900.0, 169999.0, 4.7)
('APPLE iPhone 14 Pro Max', 'Deep Purple', 149900.0, 137999.0, 4.7)
('APPLE iPhone 11 Pro Max', 'Gold', 117100.0, 95699.0, 4.7)
('APPLE iPhone 14 Pro', 'Silver', 159900.0, 149999.0, 4.7)
('APPLE iPhone 13 Pro', 'Alpine Green', 119900.0, 119900.0, 4.7)
```

```
('APPLE iPhone 13 Pro', 'Alpine Green', 169900.0, 144999.0, 4.7)
```

**Q.** SQL statement to select all the columns from the "iPhone" table where the "product\_name" column ends with the string "XS"

```
query_2 = '''SELECT * FROM iPhoneData \
            WHERE product_name LIKE '%XS' '''
for row in c.execute(query_2):
    print(row)
```

```
⇒ ('APPLE iPhone XS', 'Space Grey', '64 GB ROM', '13.97 cm (5.5 inch) Retina HD
('APPLE iPhone XS', 'Gold', '64 GB ROM', '13.97 cm (5.5 inch) Retina HD Displ
('APPLE iPhone XS', 'Space Grey', '256 GB ROM', '13.72 cm (5.4 inch) Super Re
('APPLE iPhone XS', 'Gold', '256 GB ROM', '13.72 cm (5.4 inch) Super Retina XI
('APPLE iPhone XS', 'Space Grey', '64 GB ROM', '15.49 cm (6.1 inch) Display',
('APPLE iPhone XS', 'Silver', '256 GB ROM', '11.94 cm (4.7 inch) Retina HD Di
('APPLE iPhone XS', 'Silver', '256 GB ROM', '11.94 cm (4.7 inch) Retina HD Di
('APPLE iPhone XS', 'Silver', '64 GB ROM', '15.49 cm (6.1 inch) Display', '12M
('APPLE iPhone XS', 'Gold', '64 GB ROM', '15.49 cm (6.1 inch) Display', '12MP
```

**Q.** SQL statement to select all the columns from "iPhone" table where storage is greater than or equal to 512