


```
df = pd.read_csv('orders.csv',na_values=['Not Available','unknown'])
df['Ship Mode'].unique()
```

```
array(['Second Class', 'Standard Class', nan, 'First Class', 'Same Day'],
      dtype=object)
```

```
#renaming the columns and make them lower case and replace space with underscore
```

```
df.columns=df.columns.str.lower()
df.columns=df.columns.str.replace(' ','_')
df.columns
```

```
Index(['order_id', 'order_date', 'ship_mode', 'segment', 'country', 'city',
       'state', 'postal_code', 'region', 'category', 'sub_category',
       'product_id', 'cost_price', 'list_price', 'quantity',
       'discount_percent'],
      dtype='object')
```

```
#devine new columns discount, sale_price, profit
df['discount']=df['list_price']*df['discount_percent']*0.01
df['sale_price']=df['list_price']-df['discount']
df['profit']=df['sale_price']-df['cost_price']
df.head(5)
```

```
array([[0, 1, 2023-03-01, Second Class, Consumer, United States, Henderson, Kentucky, 42420, South, Furniture, Bookcases, 1000000, 1000000, 1000000],
       [1, 2, 2023-08-15, Second Class, Consumer, United States, Henderson, Kentucky, 42420, South, Furniture, Chairs, 1000000, 1000000, 1000000],
       [2, 3, 2023-01-10, Second Class, Corporate, United States, Los Angeles, California, 90036, West, Office Supplies, Labels, 1000000, 1000000, 1000000],
       [3, 4, 2022-06-18, Standard Class, Consumer, United States, Fort Lauderdale, Florida, 33311, South, Furniture, Tables, 1000000, 1000000, 1000000],
       [4, 5, 2022-07-13, Standard Class, Consumer, United States, Fort Lauderdale, Florida, 33311, South, Office Supplies, Storage, 1000000, 1000000, 1000000]])
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9994 entries, 0 to 9993
Data columns (total 19 columns):
#   Column                Non-Null Count  Dtype
---  ---
0   order_id              9994 non-null   int64
1   order_date            9994 non-null   object
2   ship_mode             9988 non-null   object
3   segment               9994 non-null   object
4   country               9994 non-null   object
5   city                  9994 non-null   object
6   state                 9994 non-null   object
7   postal_code           9994 non-null   int64
8   region                9994 non-null   object
9   category              9994 non-null   object
10  sub_category          9994 non-null   object
11  product_id            9994 non-null   object
12  cost_price            9994 non-null   int64
13  list_price            9994 non-null   int64
14  quantity              9994 non-null   int64
15  discount_percent      9994 non-null   int64
16  discount              9994 non-null   float64
17  sale_price            9994 non-null   float64
18  profit               9994 non-null   float64
dtypes: float64(3), int64(6), object(10)
memory usage: 1.4+ MB
```

```
df['order_date']=pd.to_datetime(df['order_date'],format="%Y-%m-%d") ##converting the order_date to datetime format
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9994 entries, 0 to 9993
Data columns (total 19 columns):
#   Column                Non-Null Count  Dtype
---  ---
0   order_id              9994 non-null   int64
1   order_date            9994 non-null   datetime64[ns]
2   ship_mode             9988 non-null   object
```

```

3 segment          9994 non-null object
4 country          9994 non-null object
5 city             9994 non-null object
6 state           9994 non-null object
7 postal_code      9994 non-null int64
8 region          9994 non-null object
9 category         9994 non-null object
10 sub_category    9994 non-null object
11 product_id      9994 non-null object
12 cost_price      9994 non-null int64
13 list_price      9994 non-null int64
14 quantity        9994 non-null int64
15 discount_percent 9994 non-null int64
16 discount        9994 non-null float64
17 sale_price      9994 non-null float64
18 profit          9994 non-null float64
dtypes: datetime64[ns](1), float64(3), int64(6), object(9)
memory usage: 1.4+ MB

```

```
df.head()
```

	order_id	order_date	ship_mode	segment	country	city	state	postal_code	region	category	sub_category	product_id
0	1	2023-03-01	Second Class	Consumer	United States	Henderson	Kentucky	42420	South	Furniture	Bookcases	F10001
1	2	2023-08-15	Second Class	Consumer	United States	Henderson	Kentucky	42420	South	Furniture	Chairs	F10002
2	3	2023-01-10	Second Class	Corporate	United States	Los Angeles	California	90036	West	Office Supplies	Labels	C10003
3	4	2022-06-18	Standard Class	Consumer	United States	Fort Lauderdale	Florida	33311	South	Furniture	Tables	F10004
4	5	2022-07-13	Standard Class	Consumer	United States	Fort Lauderdale	Florida	33311	South	Office Supplies	Storage	C10005

```
df.drop(columns=['list_price','cost_price','discount_percent'],inplace=True) #dropping the columns
df.head()
```

	order_id	order_date	ship_mode	segment	country	city	state	postal_code	region	category	sub_category	product_id
0	1	2023-03-01	Second Class	Consumer	United States	Henderson	Kentucky	42420	South	Furniture	Bookcases	F10001
1	2	2023-08-15	Second Class	Consumer	United States	Henderson	Kentucky	42420	South	Furniture	Chairs	F10002
2	3	2023-01-10	Second Class	Corporate	United States	Los Angeles	California	90036	West	Office Supplies	Labels	C10003

```
df.head()
```

	order_id	order_date	ship_mode	segment	country	city	state	postal_code	region	category	sub_category	product_id
0	1	2023-03-01	Second Class	Consumer	United States	Henderson	Kentucky	42420	South	Furniture	Bookcases	F10001
1	2	2023-08-15	Second Class	Consumer	United States	Henderson	Kentucky	42420	South	Furniture	Chairs	F10002
2	3	2023-01-10	Second Class	Corporate	United States	Los Angeles	California	90036	West	Office Supplies	Labels	C10003

3. Load Data

Load data into SQL

```

#loading the data into the database
import sqlite3
conn = sqlite3.connect('orders.db')
df.to_sql('orders',conn,if_exists='replace',index=False)
conn.close()

```

```

#read the data from the database
import sqlite3

```

```
conn = sqlite3.connect('orders.db')
data = pd.read_sql('SELECT * FROM orders', conn)
data.head()
```

	order_id	order_date	ship_mode	segment	country	city	state	postal_code	region	category	sub_category	product_id
0	1	2023-03-01T00:00:00	Second Class	Consumer	United States	Henderson	Kentucky	42420	South	Furniture	Bookcases	F1000001
1	2	2023-08-15T00:00:00	Second Class	Consumer	United States	Henderson	Kentucky	42420	South	Furniture	Chairs	F1000002
2	3	2023-01-10T00:00:00	Second Class	Corporate	United States	Los Angeles	California	90036	West	Office Supplies	Labels	C1000003
3	4	2022-06-18T00:00:00	Standard Class	Consumer	United States	Fort Lauderdale	Florida	33311	South	Furniture	Tables	F1000004
4	5	2022-07-01T00:00:00	Standard Class	Consumer	United States	Fort Lauderdale	Florida	33311	South	Office Supplies	Storage	C1000005

```
#total sales, profit, discount
data = pd.read_sql('SELECT SUM(sale_price) as total_sales, SUM(profit) as total_profit, SUM(discount) as total_discount FROM orders')
data.head()
```

	total_sales	total_profit	total_discount
0	2215858.7	205168.7	80331.3

```
# find top 10 highest revenue generating products
data = pd.read_sql('SELECT product_id, SUM(sale_price) as total_sales FROM orders GROUP BY product_id ORDER BY total_sales DESC LIMIT 10')
data.head()
```

	product_id	total_sales
0	TEC-CO-10004722	59514.0
1	OFF-BI-10003527	26525.3
2	TEC-MA-10002412	21734.4
3	FUR-CH-10002024	21096.2
4	OFF-BI-10001359	19090.2

```
# find top 10 highest revenue generating cities
data = pd.read_sql('SELECT city, SUM(sale_price) as total_sales FROM orders GROUP BY city ORDER BY total_sales DESC LIMIT 10')
data.head()
```

	city	total_sales
0	New York City	247205.7
1	Los Angeles	169758.4
2	Seattle	115358.7
3	San Francisco	108890.1
4	Philadelphia	105258.3

```
# find top 5 highest selling products in each region
data = pd.read_sql('SELECT region, product_id, SUM(sale_price) as total_sales FROM orders GROUP BY region, product_id ORDER BY total_sales DESC')
data.groupby('region').head(5)
data.head()
```

	region	product_id	total_sales
0	Central	TEC-CO-10004722	16975.0
1	Central	TEC-MA-10000822	13770.0
2	Central	OFF-BI-10001120	11056.5
3	Central	OFF-BI-10000545	10132.7
4	Central	OFF-BI-10004995	8416.1

```
# for each category find the month with the highest sales
data = pd.read_sql('SELECT category, strftime("%m",order_date) as month, SUM(sale_price) as total_sales FROM orders GROUP BY category, month')
# get the highest sales month for each category
result = data.groupby('category').head(1)
```

```
print("\nHighest sales month for each category:")
print(result)
```



```
Highest sales month for each category:
      category month  total_sales
0      Furniture    08      71649.5
12 Office Supplies  02      77959.5
24    Technology   10     103021.1
```

```
# Install nbconvert if not already installed (uncomment if needed)
# !pip install nbconvert
```

```
# Convert and export notebook to PDF
!jupyter nbconvert --to pdf 'SQL_Python_Retail_Order_Analysis.ipynb'
```



```
usage: jupyter [-h] [--version] [--config-dir] [--data-dir] [--runtime-dir]
              [--paths] [--json] [--debug]
              [subcommand]
```

Jupyter: Interactive Computing

positional arguments:
subcommand the subcommand to launch

options:
-h, --help show this help message and exit
--version show the versions of core jupyter packages and exit
--config-dir show Jupyter config dir
--data-dir show Jupyter data dir
--runtime-dir show Jupyter runtime dir
--paths show all Jupyter paths. Add --json for machine-readable format.
--json output paths as machine-readable json
--debug output debug information about paths

Available subcommands: kernel kernelspec migrate run troubleshoot

Jupyter command `jupyter-nbconvert` not found.

```
pip install nbconvert
```



```
Requirement already satisfied: nbconvert in /Users/itspawarajput/Desktop/Goal/RedditDataEngineering/.venv/lib/python3.1
Requirement already satisfied: beautifulsoup4 in /Users/itspawarajput/Desktop/Goal/RedditDataEngineering/.venv/lib/pyth
Requirement already satisfied: bleach!=5.0.0 in /Users/itspawarajput/Desktop/Goal/RedditDataEngineering/.venv/lib/pytho
Requirement already satisfied: defusedxml in /Users/itspawarajput/Desktop/Goal/RedditDataEngineering/.venv/lib/python3.
Requirement already satisfied: Jinja2>=3.0 in /Users/itspawarajput/Desktop/Goal/RedditDataEngineering/.venv/lib/python3
Requirement already satisfied: jupyter-core>=4.7 in /Users/itspawarajput/Desktop/Goal/RedditDataEngineering/.venv/lib/p
Requirement already satisfied: jupyterlab-pygments in /Users/itspawarajput/Desktop/Goal/RedditDataEngineering/.venv/lib
Requirement already satisfied: markupsafe>=2.0 in /Users/itspawarajput/Desktop/Goal/RedditDataEngineering/.venv/lib/pyt
Requirement already satisfied: mistune<4,>=2.0.3 in /Users/itspawarajput/Desktop/Goal/RedditDataEngineering/.venv/lib/p
Requirement already satisfied: nbclient>=0.5.0 in /Users/itspawarajput/Desktop/Goal/RedditDataEngineering/.venv/lib/pyt
Requirement already satisfied: nbformat>=5.7 in /Users/itspawarajput/Desktop/Goal/RedditDataEngineering/.venv/lib/pytho
Requirement already satisfied: packaging in /Users/itspawarajput/Desktop/Goal/RedditDataEngineering/.venv/lib/python3.1
Requirement already satisfied: pandocfilters>=1.4.1 in /Users/itspawarajput/Desktop/Goal/RedditDataEngineering/.venv/li
Requirement already satisfied: pygments>=2.4.1 in /Users/itspawarajput/Desktop/Goal/RedditDataEngineering/.venv/lib/pyt
Requirement already satisfied: traitlets>=5.1 in /Users/itspawarajput/Desktop/Goal/RedditDataEngineering/.venv/lib/pyth
Requirement already satisfied: webencodings in /Users/itspawarajput/Desktop/Goal/RedditDataEngineering/.venv/lib/python
Requirement already satisfied: tinycss2<1.5,>=1.1.0 in /Users/itspawarajput/Desktop/Goal/RedditDataEngineering/.venv/li
Requirement already satisfied: platformdirs>=2.5 in /Users/itspawarajput/Desktop/Goal/RedditDataEngineering/.venv/lib/p
Requirement already satisfied: jupyter-client>=6.1.12 in /Users/itspawarajput/Desktop/Goal/RedditDataEngineering/.venv
Requirement already satisfied: fastjsonschema>=2.15 in /Users/itspawarajput/Desktop/Goal/RedditDataEngineering/.venv/li
Requirement already satisfied: jsonschema>=2.6 in /Users/itspawarajput/Desktop/Goal/RedditDataEngineering/.venv/lib/pyt
Requirement already satisfied: soupsieve>1.2 in /Users/itspawarajput/Desktop/Goal/RedditDataEngineering/.venv/lib/pytho
Requirement already satisfied: typing-extensions>=4.0.0 in /Users/itspawarajput/Desktop/Goal/RedditDataEngineering/.ven
Requirement already satisfied: attrs>=22.2.0 in /Users/itspawarajput/Desktop/Goal/RedditDataEngineering/.venv/lib/pytho
Requirement already satisfied: jsonschema-specifications>=2023.03.6 in /Users/itspawarajput/Desktop/Goal/RedditDataEngi
Requirement already satisfied: referencing>=0.28.4 in /Users/itspawarajput/Desktop/Goal/RedditDataEngineering/.venv/lib
Requirement already satisfied: rpds-py>=0.7.1 in /Users/itspawarajput/Desktop/Goal/RedditDataEngineering/.venv/lib/pyth
Requirement already satisfied: python-dateutil>=2.8.2 in /Users/itspawarajput/Desktop/Goal/RedditDataEngineering/.venv
Requirement already satisfied: pyzmq>=23.0 in /Users/itspawarajput/Desktop/Goal/RedditDataEngineering/.venv/lib/python3
Requirement already satisfied: tornado>=6.2 in /Users/itspawarajput/Desktop/Goal/RedditDataEngineering/.venv/lib/python
Requirement already satisfied: six>=1.5 in /Users/itspawarajput/Desktop/Goal/RedditDataEngineering/.venv/lib/python3.13
Note: you may need to restart the kernel to use updated packages.
```

Start coding or [generate](#) with AI.

