E-COMMERCE DATA

datalab

```python
# Extract date time by using python
import pandas as pd
df = pd.read_csv('retail_sales_clean.csv')

# Display the first few rows
print(df.head())

df['InvoiceDate'] = pd.to_datetime(df['InvoiceDate'], errors='coerce')

# Display the DataFrame
print(df.head())

df['Year_invoice'] = df['InvoiceDate'].dt.year
df['Month_invoice'] = df['InvoiceDate'].dt.month
df['Day_invoice'] = df['InvoiceDate'].dt.day
df['Hour_invoice'] = df['InvoiceDate'].dt.hour
df['Minute_invoice'] = df['InvoiceDate'].dt.minute
df['Second_invoice'] = df['InvoiceDate'].dt.second

# Display the DataFrame


# Save the DataFrame to a CSV file
df.to_csv('retail_sales_clean2.csv', index=False)
```

```
   InvoiceNo StockCode  ... Minute_invoice  Second_invoice
0     536365    85123A  ...              26               0
1     536365     71053  ...              26               0
2     536365    84406B  ...              26               0
3     536365    84029G  ...              26               0
4     536365    84029E  ...              26               0

[5 rows x 14 columns]
   InvoiceNo StockCode  ... Minute_invoice  Second_invoice
0     536365    85123A  ...              26               0
1     536365     71053  ...              26               0
2     536365    84406B  ...              26               0
3     536365    84029G  ...              26               0
4     536365    84029E  ...              26               0

[5 rows x 14 columns]
```

E-Commerce Data DataFrame as `variable_explanation_csv`

```sql
SELECT * FROM 'variable_explanation.csv'
```

index	Variable	Explanation
0	InvoiceNo	A 6-digit integral number uniquely assigned to each transaction. If this code starts with lette...
1	StockCode	A 5-digit integral number uniquely assigned to each distinct product.
2	Description	Product (item) name
3	Quantity	The quantities of each product (item) per transaction
4	InvoiceDate	The day and time when each transaction was generated
5	UnitPrice	Product price per unit in sterling (pound)
6	CustomerID	A 5-digit integral number uniquely assigned to each customer
7	Country	The name of the country where each customer resides

Rows: 8

retail_sales_clean2 DataFrame as `df11`

```
SELECT * FROM  retail_sales_clean2.csv;
```

	I..	Stoc...	Description		InvoiceDate	U	C...
41	536370	21913	VINTAGE SEASIDE JIGSAW PUZZLES	12	2010-12-01T08:45:00.000	3.75	
42	536370	22540	MINI JIGSAW CIRCUS PARADE	24	2010-12-01T08:45:00.000	0.42	
43	536370	22544	MINI JIGSAW SPACEBOY	24	2010-12-01T08:45:00.000	0.42	
44	536370	22492	MINI PAINT SET VINTAGE	36	2010-12-01T08:45:00.000	0.65	
45	536370	POST	POSTAGE	3	2010-12-01T08:45:00.000	18	
46	536371	22086	PAPER CHAIN KIT 50'S CHRISTMAS	80	2010-12-01T09:00:00.000	2.55	
47	536372	22632	HAND WARMER RED POLKA DOT	6	2010-12-01T09:01:00.000	1.85	
48	536372	22633	HAND WARMER UNION JACK	6	2010-12-01T09:01:00.000	1.85	
49	536373	85123A	WHITE HANGING HEART T-LIGHT HOLDER	6	2010-12-01T09:02:00.000	2.55	
50	536373	71053	WHITE METAL LANTERN	6	2010-12-01T09:02:00.000	3.39	
51	536373	84406B	CREAM CUPID HEARTS COAT HANGER	8	2010-12-01T09:02:00.000	2.75	
52	536373	20679	EDWARDIAN PARASOL RED	6	2010-12-01T09:02:00.000	4.95	
53	536373	37370	RETRO COFFEE MUGS ASSORTED	6	2010-12-01T09:02:00.000	1.06	
54	536373	21871	SAVE THE PLANET MUG	6	2010-12-01T09:02:00.000	1.06	
55	536373	21071	VINTAGE BILLBOARD DRINK ME MUG	6	2010-12-01T09:02:00.000	1.06	
56	536373	21068	VINTAGE BILLBOARD LOVE/HATE MUG	6	2010-12-01T09:02:00.000	1.06	

Rows: 7,142 ⚠ truncated from 757,442 rows

retail_sales_clean2 DataFrame as `df`

```
--Total Sales Revenue

--How much total revenue has been generated?
SELECT sum(Quantity * UnitPrice) AS total_revenue
FROM retail_sales_clean2.csv;
```

index	total_revenue
0	17576885.186007008

Rows: 1

retail_sales_clean2 DataFrame as `df1`

```
--Top-Selling Products

--What are the top 10 best-selling products by quantity?
SELECT Description,
sum(Quantity) AS sales_revenue
FROM retail_sales_clean2.csv
GROUP BY Description
ORDER BY sales_revenue DESC LIMIT 10 ;
```

index	Description	sales_revenue
0	PAPER CRAFT , LITTLE BIRDIE	161990
1	MEDIUM CERAMIC TOP STORAGE JAR	155804
2	WORLD WAR 2 GLIDERS ASSTD DESIGNS	108446
3	JUMBO BAG RED RETROSPOT	91792
4	WHITE HANGING HEART T-LIGHT HOLDER	72954
5	ASSORTED COLOUR BIRD ORNAMENT	69824
6	PACK OF 72 RETROSPOT CAKE CASES	67118
7	POPCORN HOLDER	61312
8	RABBIT NIGHT LIGHT	53984
9	MINI PAINT SET VINTAGE	52152

Rows: 10

retail_sales_clean2 DataFrame as `df2`

```
--Least-Selling Products

--What are the 10 least-sold products?
SELECT Description,
sum(Quantity) AS sales_revenue
FROM retail_sales_clean2.csv
GROUP BY Description
ORDER BY sales_revenue LIMIT 10 ;
```

index	Description	sales_revenue
0	MUMMY MOUSE RED GINGHAM RIBBON	2
1	BLUE PADDED SOFT MOBILE	2
2	LASER CUT MULTI STRAND NECKLACE	2
3	AMBER BERTIE GLASS BEAD BAG CHARM	2
4	DOLPHIN WINDMILL	2
5	I LOVE LONDON MINI RUCKSACK	2
6	SET 12 COLOURING PENCILS DOILEY	2
7	CRACKED GLAZE EARRINGS RED	2
8	BAROQUE BUTTERFLY EARRINGS CRYSTAL	2
9	POTTING SHED SOW 'N' GROW SET	2

Rows: 10

retail_sales_clean2 DataFrame as `df3`

```
--Customer Behavior Analysis
--Most Valuable Customers
--Who are the top 5 customers by total spending?
SELECT  CustomerID,
sum(Quantity*UnitPrice) AS sales_revenue
FROM retail_sales_clean2.csv
GROUP BY CustomerID
ORDER BY sales_revenue DESC LIMIT 5;
```

index	CustomerID	sales_revenue
0	14646	560412.0400000005
1	18102	519314.6000000001
2	17450	388461.5800000002
3	16446	336945
4	14911	287194.559999998

Rows: 5

retail_sales_clean2 DataFrame as `df5`

```
--Customer Order Frequency

--How many unique orders has each customer placed?

SELECT CustomerID,
count(DISTINCT InvoiceNo) AS count_order
FROM retail_sales_clean2.csv
GROUP BY CustomerID
ORDER BY count_order DESC LIMIT 10;
```

index	CustomerID	count_order
0	12748	209
1	14911	201
2	17841	124
3	13089	97
4	14606	93
5	15311	91
6	12971	86
7	14646	73
8	16029	63
9	13408	62

Rows: 10

retail_sales_clean DataFrame as `df6`

```
--Average Order Value (AOV)

--What is the average revenue per order?

SELECT
    SUM(Quantity * UnitPrice) AS sum_revenue,
    COUNT(DISTINCT InvoiceNo) AS total_order,
    SUM(Quantity * UnitPrice) / COUNT(DISTINCT InvoiceNo) AS avg_order_value
FROM retail_sales_clean2.csv;
```

index	sum_revenue	total_order	avg_order_value
0	17576885.186007053	18518	949.1783770389

Rows: 1

retail_sales_clean2 DataFrame as `df9`

```
--Geographical Analysis
--Top 5 Countries by Revenue

--Which countries generate the highest sales revenue?
SELECT Country,
sum(Quantity*UnitPrice) AS revenue_sales
FROM retail_sales_clean2.csv
GROUP BY Country
ORDER BY revenue_sales DESC LIMIT 5;
```

index	Country	revenue_sales
0	United Kingdom	14401490.686006756
1	Netherlands	570892.6800000005
2	EIRE	529572.1400000005
3	Germany	454568.3799999969
4	France	417318.0599999982

Rows: 5

retail_sales_clean2 DataFrame as `df10`

```
--Customer Distribution by Country

--How many unique customers are there in each country?

SELECT Country,
count(DISTINCT CustomerID) AS unique_customers
FROM online_retail.csv
GROUP BY Country
ORDER BY unique_customers DESC;
```

index	Country	unique_customers
0	United Kingdom	3950
1	Germany	95
2	France	87
3	Spain	31
4	Belgium	25
5	Switzerland	21
6	Portugal	19
7	Italy	15
8	Finland	12
9	Austria	11
10	Norway	10
11	Australia	9
12	Denmark	9
13	Netherlands	9
14	Channel Islands	9
15	Japan	8
16	Cyprus	8

Rows: 38

retail_sales_clean2 DataFrame as `df4`

```
--Time-Based Analysis
--Sales Trend Over Time

--What are the total sales per month?
Select Month_invoice,
SUM(Quantity*UnitPrice) AS total_revenue
FROM retail_sales_clean2.csv
GROUP BY Month_invoice
ORDER BY Month_invoice;
```

index	Month_invoice	total_revenue
0	1	1127053.5199999402
1	2	885169.4199999708
2	3	1178683.8800000227
3	4	923454.1219999763
4	5	1346958.619999952
5	6	1305396.9799999564
6	7	1191101.5419999869
7	8	1278724.1999999841
8	9	1887434.681999928
9	10	2033053.0999997891
10	11	2269048.639999788
11	12	2150806.479999904

Rows: 12

retail_sales_clean2 DataFrame as `df7`

```
--Peak Sales Hours

--What are the peak hours for purchases?
SELECT DISTINCT(Hour_invoice) AS Hour,
count(Quantity) AS count_order
FROM retail_sales_clean2.csv
GROUP BY Hour
ORDER BY count_order DESC LIMIT 5;
```

index	Hour	count_order
0	12	135564
1	13	120596
2	14	102108
3	11	92930
4	15	86384

Rows: 5

retail_sales_clean2 DataFrame as `retail_sales_clean2_csv`

```
SELECT
sum(Quantity*UnitPrice)/count(DISTINCT InvoiceNo) AS AOG
FROM 'retail_sales_clean2.csv'
```

index	AOG
0	949.1783770389

Rows: 1