SQL_Operations

> 1. Read superstore data

```
data=spark.read.csv("dbfs:/FileStore/superstore.csv",header=True, inferSchema=True)
display(data.head(5))
```

no	order_id	Order Date	Ship Date	Ship Mode	Customer ID	Customer Name	Segment	Country	City	Sta
null	null	2016-11-08	2016-11-11	Second Class	CG-12520	Claire Gute	Consumer	United States	Henderson	Ke
2	CA-2016-152156	2016-11-08	2016-11-11	Second Class	CG-12520	Claire Gute	Consumer	United States	Henderson	Ke

> 2. Create a view of the DatafRame and then add it as a table to sql.

```
df3 = data.toDF(*[c.replace(" ", "_").replace(",", "").replace("(", "").replace(")", "") for

df3.createOrReplaceTempView("superstore")

%sql
--create table if not exists superstore as
--select*from superstore;

%sql
select*from superstore limit 5;
```

no	order__id	Order_Date	Ship_Date	Ship_Mode	Customer_ID	Customer_Name	Segment
null	null	2016-11-08	2016-11-11	Second Class	CG-12520	Claire Gute	Consumer
2	CA-2016-152156	2016-11-08	2016-11-11	Second Class	CG-12520	Claire Gute	Consumer
3	CA-2016-138688	2016-06-12	2016-06-16	Second Class	DV-13045	Darrin Van Huff	Corporate

3. find the columns of the data

```
%sql
desc superstore;
```

col_name	data_type	comment
no	int	null
order__id	string	null
Order_Date	date	null
Ship_Date	date	null
Ship_Mode	string	null
Customer_ID	string	null
Customer_Name	string	null
Segment	string	null
Country	string	null
City	string	null

4. display first 40 records

```
%sql
select*from superstore limit 40;
```

no	order__id	Order_Date	Ship_Date	Ship_Mode	Customer_ID	Customer_Name	Segment
null	null	2016-11-08	2016-11-11	Second Class	CG-12520	Claire Gute	Consumer
2	CA-2016-152156	2016-11-08	2016-11-11	Second Class	CG-12520	Claire Gute	Consumer
3	CA-2016-138688	2016-06-12	2016-06-16	Second Class	DV-13045	Darrin Van Huff	Corporate

5. display last 15 records from the data

```
%sql
select * from superstore
order by no desc
limit 15;
```

no	order__id	Order_Date	Ship_Date	Ship_Mode	Customer_ID	Customer_Name	Segment
9994	CA-2017-119914	2017-05-04	2017-05-09	Second Class	CC-12220	Chris Cortes	Consume
9993	CA-2017-121258	2017-02-26	2017-03-03	Standard Class	DB-13060	Dave Brooks	Consume
9992	CA-2017-	2017-03-	Standard				

6. take a slice from the dataframe based on the columns city, state, region, profit columns

```
%sql
select city, state, region, profit
```

```
from superstore;
```

city	state	region	profit
Henderson	Kentucky	South	41.9136
Henderson	Kentucky	South	219.582
Los Angeles	California	West	6.8714
Fort Lauderdale	Florida	South	-383.031
Fort Lauderdale	Florida	South	2.5164
Los Angeles	California	West	14.1694
Los Angeles	California	West	1.9656
Los Angeles	California	West	90.7152
Los Angeles	California	West	5.7825
Los Angeles	California	West	34.47

7. Take a slice based on data available at 1000 to 1500 position and columns from 4 to 8

```
%sql
select Ship_Date,Ship_Mode,Customer_ID,Customer_Name,Segment
from superstore
where no between 1000 and 1500;
```

Ship_Date	Ship_Mode	Customer_ID	Customer_Name	Segment
2015-11-03	Standard Class	RD-19585	Rob Dowd	Consumer
2016-11-17	Standard Class	FM-14290	Frank Merwin	Home Office
2015-07-31	Same Day	RH-19495	Rick Hansen	Consumer
2015-07-31	Same Day	RH-19495	Rick Hansen	Consumer
2015-07-31	Same Day	RH-19495	Rick Hansen	Consumer
2015-08-31	Standard Class	RA-19945	Ryan Akin	Consumer
2015-11-17	Standard Class	JK-15730	Joe Kamberova	Consumer
2015-11-09	First Class	MT-17815	Meg Tillman	Consumer
2015-11-09	First Class	MT-17815	Meg Tillman	Consumer
2018-01-01	Standard Class	PO-18850	Patrick O'Brill	Consumer

8. Take a slice from data based on index having name as 1920 , 1940, 1945, 1980, and

columns as profit, state, category, sub-category, region

```
%sql
select profit, state, category,`Sub-Category`, region
from superstore
where no in (1920,1940,1945,1980);
```

profit	state	category	Sub-Category	region
302.373	New York	Office Supplies	Binders	East
-6.1152	Colorado	Office Supplies	Binders	West
26.3912	Texas	Technology	Accessories	Central
105.0228	Arizona	Furniture	Chairs	West

9. Find out the record for the month of jun-2014

```
%sql
select * from superstore
where (Ship_Date >= '2014-06-01' AND Ship_Date <= '2014-06-30')
or (Order_Date >= '2014-06-01' AND Order_Date <= '2014-06-30');
```

no	order__id	Order_Date	Ship_Date	Ship_Mode	Customer_ID	Customer_Name	Segment
6	CA-2014-115812	2014-06-09	2014-06-14	Standard Class	BH-11710	Brosina Hoffman	Consume
7	CA-2014-115812	2014-06-09	2014-06-14	Standard Class	BH-11710	Brosina Hoffman	Consume
8	CA-2014-115812	2014-06-09	2014-06-14	Standard Class	BH-11710	Brosina Hoffman	Consume
9	CA-2014-115812	2014-06-09	2014-06-14	Standard Class	BH-11710	Brosina Hoffman	Consume

10. Calculate month wise average of profit.

```
%sql
select month(Order_Date) as Months,avg(profit) from superstore
group by month(Order_Date)
order by month(Order_Date) asc;
```

Months	avg(profit)
1	23.978425984251963
2	33.98977466666665
3	40.74613347701152
4	17.738469311377248
5	30.444503401360542
6	29.744987447698772
7	19.619705633802788
8	30.80149929178471
9	26.22573022415043
10	38.70966788766793

11.Create the new data frame for California

```
%sql
select*from superstore
where State='California';
```

no	order__id	Order_Date	Ship_Date	Ship_Mode	Customer_ID	Customer_Name	Segment
3	CA-2016-138688	2016-06-12	2016-06-16	Second Class	DV-13045	Darrin Van Huff	Corporate
6	CA-2014-115812	2014-06-09	2014-06-14	Standard Class	BH-11710	Brosina Hoffman	Consume
7	CA-2014-115812	2014-06-09	2014-06-14	Standard Class	BH-11710	Brosina Hoffman	Consume
8	CA-2014-115812	2014-06-09	2014-06-14	Standard Class	BH-11710	Brosina Hoffman	Consume

```
display(spark.sql("select*from superstore where State='California';"))
```

no	order__id	Order_Date	Ship_Date	Ship_Mode	Customer_ID	Customer_Name	Segment
3	CA-2016-138688	2016-06-12	2016-06-16	Second Class	DV-13045	Darrin Van Huff	Corporate
6	CA-2014-115812	2014-06-09	2014-06-14	Standard Class	BH-11710	Brosina Hoffman	Consume
7	CA-2014-115812	2014-06-09	2014-06-14	Standard Class	BH-11710	Brosina Hoffman	Consume
8	CA-2014-115812	2014-06-09	2014-06-14	Standard Class	BH-11710	Brosina Hoffman	Consume

12. Find sum of sales

```
%sql
select sum(Sales)as Sum_of_Sales from superstore;
```

Sum_of_Sales
2272449.8562999545

13. Find state wise sum of sales.

```
%sql
select State,sum(Sales)as TotalSales from superstore
group by State
order by State;
```

State	TotalSales
Alabama	19510.639999999992
Arizona	34284.905000000006
Arkansas	11495.189999999997
California	450567.5915000007
Colorado	31285.235999999997
Connecticut	13317.796999999995
Delaware	27036.168999999994
District of Columbia	2865.0199999999995
Florida	88876.883
Georgia	48296.30000000001

14. Find month wise sum of sales

```
%sql
select month(Order_Date) as Months,sum(Sales)as TotalSales from superstore
group by month(Order_Date)
order by Months asc;
```

Months	TotalSales
1	94539.34159999999
2	59012.82540000001
3	203719.26379999987
4	135387.35759999996
5	153513.3096999999
6	151039.43330000006
7	145623.8500000002
8	157642.25099999984
9	303536.6657000002
10	198440.0027000002

15. Find country wise and city wise average profit

```
%sql
select Country,City,avg(Profit)as Average_Profit from superstore
group by Country,City;
```

Country	City	Average_Profit
United States	Huntington Beach	146.67319999999998
United States	Carol Stream	-7.9736600000000255
United States	Indianapolis	119.20233913043478
United States	Springdale	1.419
United States	Albuquerque	45.29200714285714
United States	Thomasville	11.131299999999998
United States	Minneapolis	296.72106956521736
United States	Carlsbad	14.34237272727273
United States	Monroe	33.172580952380954
United States	Rockville	26.64294

16. Calculate region wise average profit.

```
%sql
select Region,avg(Profit)as Average_Profit from superstore
group by Region;
```

Region	Average_Profit
South	28.79650679012348
Central	17.28390142057683
East	32.16399462780904
West	33.500999531689054

17.Find the ship mode which is more profitable.

```
%sql
select ship_mode, max(profit) as profit
from superstore
group by ship_mode
order by profit desc
limit 1;
```

ship_mode	profit
Standard Class	8399.976

18. Create a new data frame where loss is recorded

```
%sql
select*from superstore
where Profit<0;
```

no	order__id	Order_Date	Ship_Date	Ship_Mode	Customer_ID	Customer_Name	Segment
4	US-2015-108966	2015-10-11	2015-10-18	Standard Class	SO-20335	Sean O'Donnell	Consume
15	US-2015-118983	2015-11-22	2015-11-26	Standard Class	HP-14815	Harold Pawlan	Home Office
16	US-2015-118983	2015-11-22	2015-11-26	Standard Class	HP-14815	Harold Pawlan	Home Office
24	US-2017-	2017-07-16	2017-07-	Second	SF-20065	Sandra Flanagan	Consume

```
display(spark.sql("select*from superstore where Profit<0"))
```

no	order__id	Order_Date	Ship_Date	Ship_Mode	Customer_ID	Customer_Name	Segment
4	US-2015-108966	2015-10-11	2015-10-18	Standard Class	SO-20335	Sean O'Donnell	Consume
15	US-2015-118983	2015-11-22	2015-11-26	Standard Class	HP-14815	Harold Pawlan	Home Office
16	US-2015-118983	2015-11-22	2015-11-26	Standard Class	HP-14815	Harold Pawlan	Home Office
24	US-2017-	2017-07-16	2017-07-	Second	SF-20065	Sandra Flanagan	Consume

19. Find category wise sub category wise sales

```
%sql
select Category,`Sub-Category`,sum(sales) as total_sales
from superstore
```

```
group by category,`Sub-Category`
order by category,`Sub-Category`;
```

Category	Sub-Category	total_sales
Furniture	Bookcases	114879.99629999997
Furniture	Chairs	328449.10300000076
Furniture	Furnishings	82752.23000000004
Furniture	Tables	206965.5320000001
Office Supplies	Appliances	107532.161
Office Supplies	Art	27118.791999999954
Office Supplies	Binders	199905.71700000006
Office Supplies	Envelopes	15339.489999999993
Office Supplies	Fasteners	3008.6559999999995
Office Supplies	Labels	12486.312

20. Create a data frame for record where discount is 0

```
%sql
select*from superstore
where Discount>=0 and Discount<0.1;
```

no	order__id	Order_Date	Ship_Date	Ship_Mode	Customer_ID	Customer_Name	Segment
null	null	2016-11-08	2016-11-11	Second Class	CG-12520	Claire Gute	Consume
2	CA-2016-152156	2016-11-08	2016-11-11	Second Class	CG-12520	Claire Gute	Consume
3	CA-2016-138688	2016-06-12	2016-06-16	Second Class	DV-13045	Darrin Van Huff	Corporate
6	CA-2014-115812	2014-06-09	2014-06-14	Standard Class	BH-11710	Brosina Hoffman	Consume

```
display(spark.sql("select*from superstore where Discount>=0 and Discount<0.1"))
```

no	order__id	Order_Date	Ship_Date	Ship_Mode	Customer_ID	Customer_Name	Segment
null	null	2016-11-08	2016-11-11	Second Class	CG-12520	Claire Gute	Consume
2	CA-2016-152156	2016-11-08	2016-11-11	Second Class	CG-12520	Claire Gute	Consume
3	CA-2016-138688	2016-06-12	2016-06-16	Second Class	DV-13045	Darrin Van Huff	Corporate
6	CA-2014-115812	2014-06-09	2014-06-14	Standard Class	BH-11710	Brosina Hoffman	Consume

Dataframe_Operations

1. Read superstore data

```
display(data)
```

no	order_id	Order Date	Ship Date	Ship Mode	Customer ID	Customer Name	Segment	Country	City
null	null	2016-11-08	2016-11-11	Second Class	CG-12520	Claire Gute	Consumer	United States	Henderson
2	CA-2016-152156	2016-11-08	2016-11-11	Second Class	CG-12520	Claire Gute	Consumer	United States	Henderson
3	CA-2016-138688	2016-06-12	2016-06-16	Second Class	DV-13045	Darrin Van Huff	Corporate	United States	Los Angel
4	US-2015-108966	2015-10-11	2015-10-18	Standard Class	SO-20335	Sean O'Donnell	Consumer	United States	Fort Lauderdal

2. find the index of the data

3. find the columns of the data

```
data.columns
```

```
Out[43]: ['no',
 'order _id',
 'Order Date',
 'Ship Date',
 'Ship Mode',
 'Customer ID',
 'Customer Name',
 'Segment',
 'Country',
 'City',
 'State',
 'Postal Code',
 'Region',
 'Product ID',
 'Category',
 'Sub-Category',
 'Product Name',
 'Sales',
 'Quantity',
 'Discount',
 'Profit']
```

4. if columns name is having space then remove the space and rename the column.

find the various ways to rename the column

```
df3 = data.toDF(*[c.replace(" ", "_").replace(",", "").replace("(", "").replace(")", "") for
display(df3.head(5))
```

no	order__id	Order_Date	Ship_Date	Ship_Mode	Customer_ID	Customer_Name	Segment
null	null	2016-11-08	2016-11-11	Second Class	CG-12520	Claire Gute	Consumer
2	CA-2016-152156	2016-11-08	2016-11-11	Second Class	CG-12520	Claire Gute	Consumer
3	CA-2016-138688	2016-06-12	2016-06-16	Second Class	DV-13045	Darrin Van Huff	Corporate

5. display first 40 records

```
display(data.head(40))
```

no	order _id	Order Date	Ship Date	Ship Mode	Customer ID	Customer Name	Segment	Country	City	St
null	null	2016-11-08	2016-11-11	Second Class	CG-12520	Claire Gute	Consumer	United States	Henderson	Ke
2	CA-2016-152156	2016-11-08	2016-11-11	Second Class	CG-12520	Claire Gute	Consumer	United States	Henderson	Ke
3	CA-2016-138688	2016-06-12	2016-06-16	Second Class	DV-13045	Darrin Van Huff	Corporate	United States	Los Angeles	Ca

6. display last 15 records from the data

```
display(data.tail(15))
```

no	order _id	Order Date	Ship Date	Ship Mode	Customer ID	Customer Name	Segment	Country	City
9980	US-2016-103674	2016-12-06	2016-12-10	Standard Class	AP-10720	Anne Pryor	Home Office	United States	Los Angele
9981	US-2015-151435	2015-09-06	2015-09-09	Second Class	SW-20455	Shaun Weien	Consumer	United States	Lafayette
9982	CA-2017-163566	2017-08-03	2017-08-06	First Class	TB-21055	Ted Butterfield	Consumer	United States	Fairfield

7. change the setting to display minimum 30 rows for given data

8. change the display setting to minimum row as 3

> 9.take a slice from the dataframe based on the columns city, state, region, profit columns

```
slice=data[['City','State','Region','Profit']]
display(slice)
```

City	State	Region	Profit
Henderson	Kentucky	South	41.9136
Henderson	Kentucky	South	219.582
Los Angeles	California	West	6.8714
Fort Lauderdale	Florida	South	-383.031
Fort Lauderdale	Florida	South	2.5164
Los Angeles	California	West	14.1694
Los Angeles	California	West	1.9656
Los Angeles	California	West	90.7152
Los Angeles	California	West	5.7825
Los Angeles	California	West	34.47

> 10. Take a slice based on data available at 1000 to 1500 position and columns from 4 to 8

```
data_subset = data.filter((data.no >= 1000) & (data.no <= 1500))
slice=data_subset.select(data.columns[4:8])
display(slice)
```

Ship Mode	Customer ID	Customer Name	Segment
Standard Class	RD-19585	Rob Dowd	Consumer
Standard Class	FM-14290	Frank Merwin	Home Office
Same Day	RH-19495	Rick Hansen	Consumer
Same Day	RH-19495	Rick Hansen	Consumer
Same Day	RH-19495	Rick Hansen	Consumer
Standard Class	RA-19945	Ryan Akin	Consumer
Standard Class	JK-15730	Joe Kamberova	Consumer
First Class	MT-17815	Meg Tillman	Consumer
First Class	MT-17815	Meg Tillman	Consumer
Standard Class	PO-18850	Patrick O'Brill	Consumer

> 11. Rename the index starts from 1900 to end of the data

> 12. Take a slice from data based on index having name as 1920 , 1940, 1945, 1980, and

columns as profit, state, category, sub-category, region

```
data_subset = data.filter((data.no == 1920) | (data.no == 1940) | (data.no == 1945) | (data.
slice = data_subset.select('profit', 'state', 'category', 'sub-category')
display(slice)
```

profit	state	category	sub-category
302.373	New York	Office Supplies	Binders
-6.1152	Colorado	Office Supplies	Binders
26.3912	Texas	Technology	Accessories
105.0228	Arizona	Furniture	Chairs

> 13. Check the data type of columns

```
data.printSchema()
```

```
root
 |-- no: integer (nullable = true)
 |-- order _id: string (nullable = true)
 |-- Order Date: date (nullable = true)
 |-- Ship Date: date (nullable = true)
 |-- Ship Mode: string (nullable = true)
 |-- Customer ID: string (nullable = true)
 |-- Customer Name: string (nullable = true)
 |-- Segment: string (nullable = true)
 |-- Country: string (nullable = true)
 |-- City: string (nullable = true)
 |-- State: string (nullable = true)
 |-- Postal Code: integer (nullable = true)
 |-- Region: string (nullable = true)
 |-- Product ID: string (nullable = true)
 |-- Category: string (nullable = true)
 |-- Sub-Category: string (nullable = true)
 |-- Product Name: string (nullable = true)
 |-- Sales: string (nullable = true)
 |-- Quantity: string (nullable = true)
 |-- Discount: string (nullable = true)
 |-- Profit: double (nullable = true)
```

```
data.dtypes
```

```
Out[51]: [('no', 'int'),
 ('order _id', 'string'),
 ('Order Date', 'date'),
 ('Ship Date', 'date'),
 ('Ship Mode', 'string'),
 ('Customer ID', 'string'),
 ('Customer Name', 'string'),
 ('Segment', 'string'),
 ('Country', 'string'),
 ('City', 'string'),
 ('State', 'string'),
 ('Postal Code', 'int'),
 ('Region', 'string'),
 ('Product ID', 'string'),
 ('Category', 'string'),
 ('Sub-Category', 'string'),
 ('Product Name', 'string'),
 ('Sales', 'string'),
 ('Quantity', 'string'),
 ('Discount', 'string'),
 ('Profit', 'double')]
```

14. Find out the record for the month of jun-2014

```
from pyspark.sql.functions import to_date
data = data.withColumn("Order Date", to_date(data["Order Date"], "yyyy-MM-dd"))
data = data.withColumn("Ship Date", to_date(data["Ship Date"], "yyyy-MM-dd"))
data_subset = data.filter(
    ((data["Order Date"] >= '2014-06-01') & (data["Order Date"] <= '2014-06-30')) &
    ((data["Ship Date"] >= '2014-06-01') & (data["Ship Date"] <= '2014-06-30'))
)
display(data_subset)
```

no	order_id	Order Date	Ship Date	Ship Mode	Customer ID	Customer Name	Segment	Country	City
6	CA-2014-115812	2014-06-09	2014-06-14	Standard Class	BH-11710	Brosina Hoffman	Consumer	United States	Los Angeles
7	CA-2014-115812	2014-06-09	2014-06-14	Standard Class	BH-11710	Brosina Hoffman	Consumer	United States	Los Angeles
8	CA-2014-115812	2014-06-09	2014-06-14	Standard Class	BH-11710	Brosina Hoffman	Consumer	United States	Los Angeles
	CA-								

15. Calculate month wise average of profit.

```
from pyspark.sql import functions as F
data = data.withColumn('Order Date', F.to_date(data['Order Date'], 'yyyy-MM-dd'))
data = data.withColumn('month', F.month(data['Order Date']))
data = data.withColumn('profit', data['Profit'])
result = data.groupBy('month').agg(F.avg('profit').alias('avg_profit'))
result = result.orderBy(F.asc('month'))
display(result)
```

month	avg_profit
1	23.978425984251963
2	33.98977466666665
3	40.74613347701152
4	17.738469311377248
5	30.444503401360542
6	29.744987447698772
7	19.619705633802788
8	30.80149929178471
9	26.22573022415043
10	38.70966788766793

16. Create the new data frame for California

```
data_subset = data.filter(data.State=='California')
display(data_subset)
```

no	order_id	Order Date	Ship Date	Ship Mode	Customer ID	Customer Name	Segment	Country	City
3	CA-2016-138688	2016-06-12	2016-06-16	Second Class	DV-13045	Darrin Van Huff	Corporate	United States	Los Angel
6	CA-2014-115812	2014-06-09	2014-06-14	Standard Class	BH-11710	Brosina Hoffman	Consumer	United States	Los Angel
7	CA-2014-115812	2014-06-09	2014-06-14	Standard Class	BH-11710	Brosina Hoffman	Consumer	United States	Los Angel
	CA-	2014-	2014-	Standard		Brosina		United	

> 17. Find sum of sales

```
total_sales = data.agg(F.sum('Sales').alias('Total Sales'))
display(total_sales)
```

Total Sales
2272449.8562999545

> 18. Find state wise sum of sales.

```
total_sales = data.groupBy('State').agg(F.sum('Sales').alias('Total Sales')).orderBy('State'
display(total_sales)
```

State	Total Sales
Alabama	19510.639999999992
Arizona	34284.905000000006
Arkansas	11495.189999999997
California	450567.5915000007
Colorado	31285.235999999997
Connecticut	13317.796999999995
Delaware	27036.168999999994
District of Columbia	2865.0199999999995
Florida	88876.883
Georgia	48296.30000000001

19. Find month wise sum of sales

```
from pyspark.sql import functions as F
data = data.withColumn('month', F.month(F.col('Order Date')))
total_sales = data.groupBy('month').agg(F.sum('Sales').alias('Total Sales')).orderBy('month'
display(total_sales)
```

month	Total Sales
1	94539.34159999999
2	59012.82540000001
3	203719.26379999987
4	135387.35759999996
5	153513.3096999999
6	151039.43330000006
7	145623.8500000002
8	157642.25099999984
9	303536.6657000002
10	198440.0027000002

20. Find country wise and city wise average profit

```
total_sales = data.groupBy('Country','City').agg(F.avg('Profit').alias('Average Profit')).or
display(total_sales)
```

Country	City	Average Profit
United States	Aberdeen	6.63
United States	Abilene	-3.7584
United States	Akron	-8.887409523809529
United States	Albuquerque	45.29200714285714
United States	Alexandria	18.809193750000006
United States	Allen	-9.969375000000001
United States	Allentown	-32.350057142857146
United States	Altoona	1.4522
United States	Amarillo	-38.79683
United States	Anaheim	45.187503703703705

21. Calculate region wise average profit.

```
total_sales = data.groupBy('Region').agg(F.avg('Profit').alias('Average Profit')).orderBy('F
display(total_sales)
```

Region	Average Profit
Central	17.28390142057683
East	32.16399462780904
South	28.79650679012348
West	33.500999531689054

22. Find the ship mode which is more profitable.

```
from pyspark.sql import functions as F
total_sales = data.groupBy('Ship Mode').agg(F.sum('Profit').alias('Total Profit'))
max_profit = total_sales.agg(F.max('Total Profit')).collect()[0][0]
max_profit_mode = total_sales.filter(F.col('Total Profit') == max_profit)
display(max_profit_mode)
```

23. Create a new data frame where loss is recorded

```
loss = data.filter(F.col("Profit") < 0)
display(loss)
```

no	order_id	Order Date	Ship Date	Ship Mode	Customer ID	Customer Name	Segment	Country	City
4	US-2015-108966	2015-10-11	2015-10-18	Standard Class	SO-20335	Sean O'Donnell	Consumer	United States	Fort Lauderdale
15	US-2015-118983	2015-11-22	2015-11-26	Standard Class	HP-14815	Harold Pawlan	Home Office	United States	Fort Worth
16	US-2015-118983	2015-11-22	2015-11-26	Standard Class	HP-14815	Harold Pawlan	Home Office	United States	Fort Worth