# databricksPySpark Project Sales analysis

(https://databricks.com)

```
/FileStore/tables/sales_csv.txt

/FileStore/tables/menu_csv.txt
NameError: name 'FileStore' is not defined
```

## Sales DataFrame

```
from pyspark.sql.types import StructType,StructField,IntegerType,StringType,DateType

schema=StructType([
    StructField("product_id",IntegerType(),True),
    StructField("customer_id",StringType(),True),
    StructField("order_date",DateType(),True),
    StructField("location",StringType(),True),
    StructField("source_order",StringType(),True),
])

sales_df=spark.read.format("csv").option("inferschema","true").schema(schema).load("/FileStore/tables/sales_csv.txt")
display(sales_df)
```

Table

| | product_id | customer_id | order_date | location | source_order |
|---|---|---|---|---|---|
| 1 | 1 | A | 2023-01-01 | India | Swiggy |
| 2 | 2 | A | 2022-01-01 | India | Swiggy |
| 3 | 2 | A | 2023-01-07 | India | Swiggy |
| 4 | 3 | A | 2023-01-10 | India | Restaurant |
| 5 | 3 | A | 2022-01-11 | India | Swiggy |
| 6 | 3 | A | 2023-01-11 | India | Restaurant |
| 7 | 2 | B | 2022-02-01 | India | Swiggy |

117 rows

## deriving year month, quarter

```
from pyspark.sql.functions import month, year, quarter

sales_df=sales_df.withColumn("order_year",year(sales_df.order_date))
display(sales_df)
```

Table

| | product_id | customer_id | order_date | location | source_order | order_year |
|---|---|---|---|---|---|---|
| 1 | 1 | A | 2023-01-01 | India | Swiggy | 2023 |
| 2 | 2 | A | 2022-01-01 | India | Swiggy | 2022 |
| 3 | 2 | A | 2023-01-07 | India | Swiggy | 2023 |
| 4 | 3 | A | 2023-01-10 | India | Restaurant | 2023 |
| 5 | 3 | A | 2022-01-11 | India | Swiggy | 2022 |
| 6 | 3 | A | 2023-01-11 | India | Restaurant | 2023 |
| 7 | 2 | B | 2022-02-01 | India | Swiggy | 2022 |

117 rows

```
sales_df=sales_df.withColumn("order_month",month(sales_df.order_date))
sales_df=sales_df.withColumn("order_quarter",quarter(sales_df.order_date))
display(sales_df)
```

Table

| | product_id | customer_id | order_date | location | source_order | order_year | order_month | order_quarter |
|---|---|---|---|---|---|---|---|---|
| 1 | 1 | A | 2023-01-01 | India | Swiggy | 2023 | 1 | 1 |
| 2 | 2 | A | 2022-01-01 | India | Swiggy | 2022 | 1 | 1 |
| 3 | 2 | A | 2023-01-07 | India | Swiggy | 2023 | 1 | 1 |
| 4 | 3 | A | 2023-01-10 | India | Restaurant | 2023 | 1 | 1 |
| 5 | 3 | A | 2022-01-11 | India | Swiggy | 2022 | 1 | 1 |
| 6 | 3 | A | 2023-01-11 | India | Restaurant | 2023 | 1 | 1 |
| 7 | 2 | B | 2022-02-01 | India | Swiggy | 2022 | 2 | 1 |

117 rows

# menu dataframe

```
from pyspark.sql.types import StructType,StructField,IntegerType,StringType,DateType

schema=StructType([
    StructField("product_id",IntegerType(),True),
    StructField("product_name",StringType(),True),
    StructField("price",StringType(),True),
])

menu_df=spark.read.format("csv").option("inferschema","true").schema(schema).load("/FileStore/tables/menu_csv.txt")
display(menu_df)
```

Table

| | product_id | product_name | price |
|---|---|---|---|

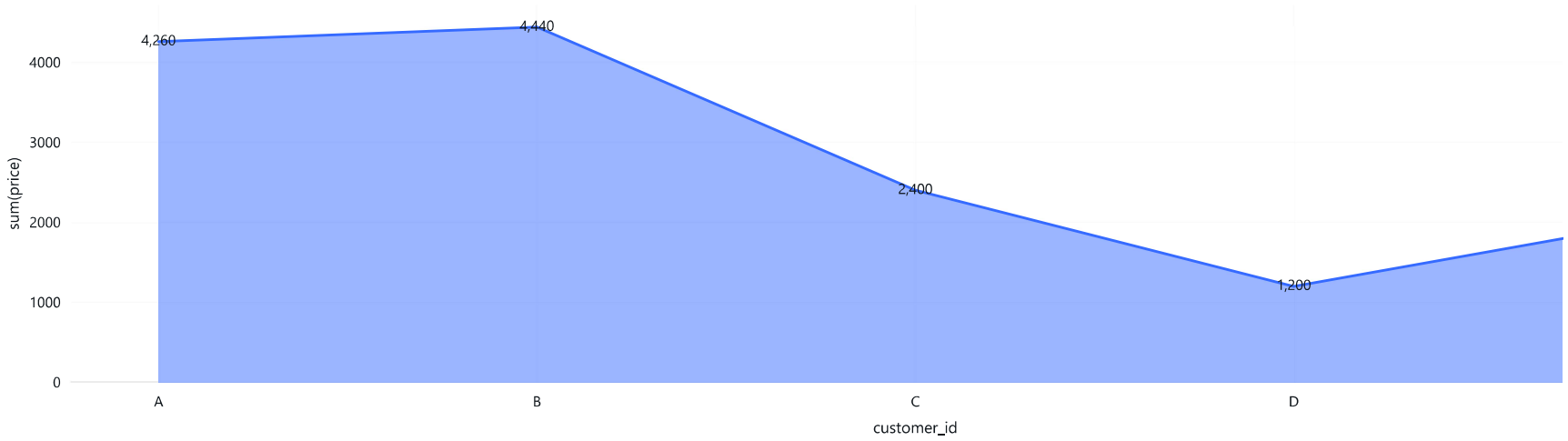
| | | | |
|---|---|---|---|
| 1 | 1 | PIZZA | 100 |
| 2 | 2 | Chowmin | 150 |
| 3 | 3 | sandwich | 120 |
| 4 | 4 | Dosa | 110 |
| 5 | 5 | Biryani | 80 |
| 6 | 6 | Pasta | 180 |

6 rows

# Total Amount spent by each customer

```
total_amount_spent = (sales_df.join(menu_df,'product_id').groupBy('customer_id').agg({'price':'sum'}).orderBy('customer_id'))
display(total_amount_spent)
```

Table **Visualization 1**

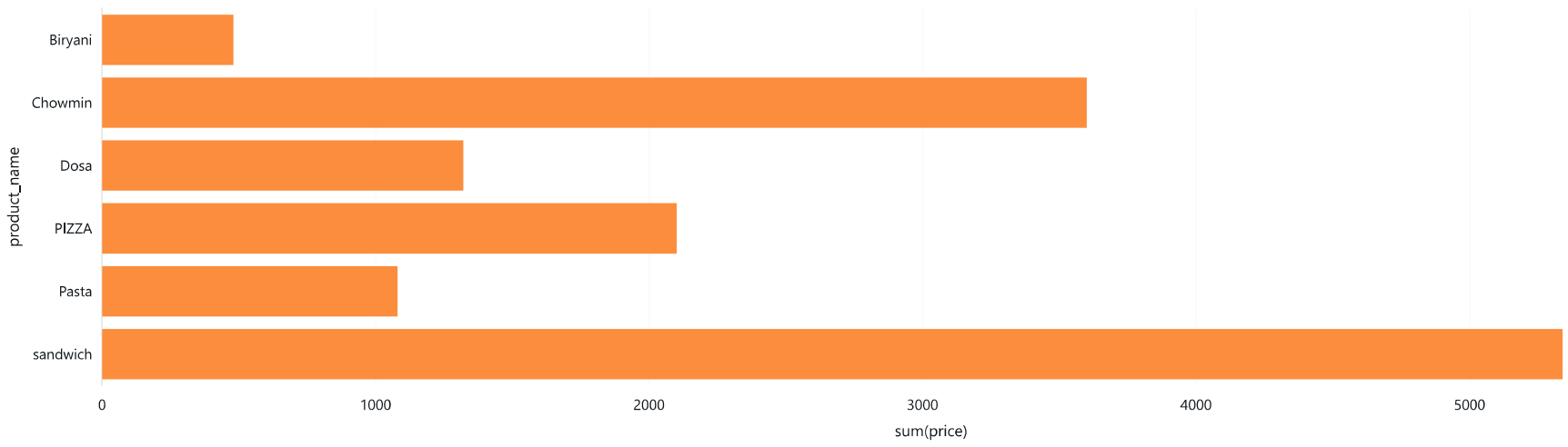



5 rows

# Total amount spent by each food Category

```
total_amount_spent= (sales_df.join(menu_df,'product_id').groupBy('product_name')
                     .agg({'price':'sum'})
                     .orderBy('product_name'))
display(total_amount_spent)
```

Table **Visualization 1**

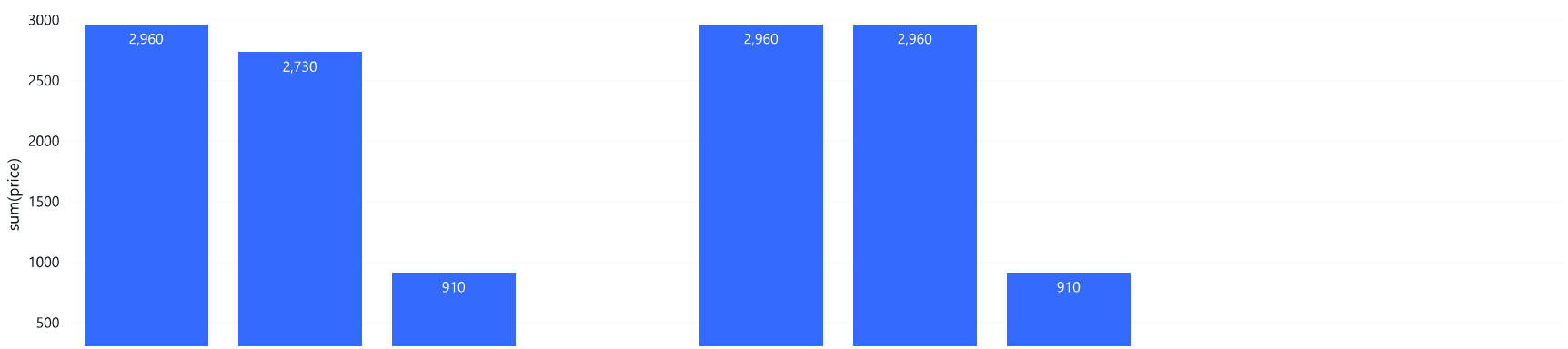



6 rows

# Total Amount of sales in each month

```
df1= (sales_df.join(menu_df,'product_id').groupBy('order_month')
                    .agg({'price':'sum'})
                    .orderBy('order_month'))
display(df1)
```

Table **Visualization 1**

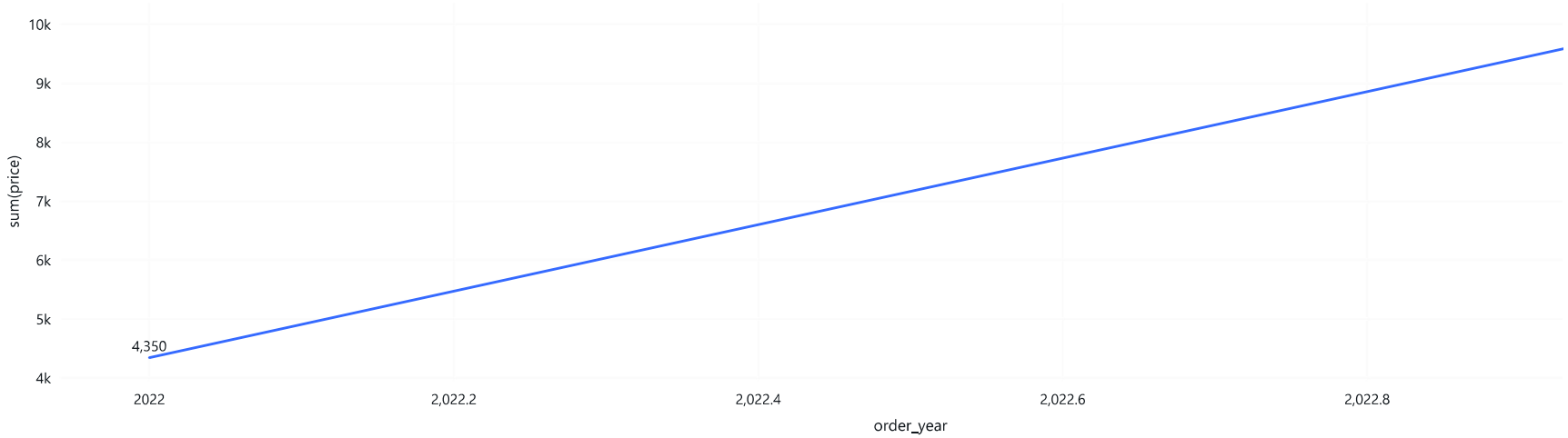



7 rows

## Yearly Sales

```
df2= (sales_df.join(menu_df,'product_id').groupBy('order_year')
                    .agg({'price':'sum'})
                    .orderBy('order_year'))
display(df2)
```

Table **Visualization 1**

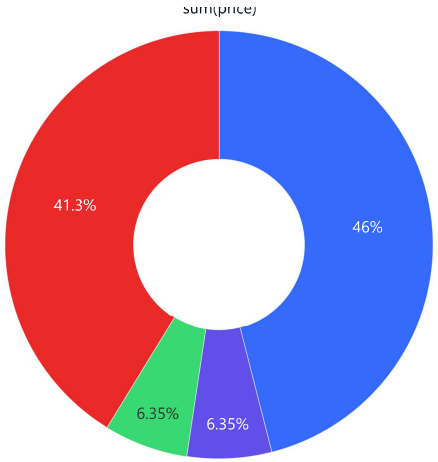



2 rows

## Quaterly Sales

```
df2= (sales_df.join(menu_df,'product_id').groupBy('order_quarter')
                    .agg({'price':'sum'})
                    .orderBy('order_quarter'))
display(df2)
```

Table **Visualization 1**




4 rows

# how many times each product purchased

```
from pyspark.sql.functions import count

most_df=(sales_df.join(menu_df,'product_id').groupBy('product_id','product_name')
         .agg(count('product_id').alias('product_count')).
         orderBy('product_count',ascending=0)
         .drop('product_id')

         )
display(most_df)
```

Table **Visualization 1**

| Steps | Value | % Max | % Previous |
|---|---|---|---|
| sandwich | 48 | 100% | 100% |
| Chowmin | 24 | 50% | 50% |
| PIZZA | 21 | 43.75% | 87.50% |

| Steps | Value | % Max | % Previous |
|---|---|---|---|
| Dosa | 12 | 25% | 57.14% |

6 rows

# Top 5 ordered items

```
from pyspark.sql.functions import count

most_df=(sales_df.join(menu_df,'product_id').groupBy('product_id','product_name')
         .agg(count('product_id').alias('product_count')).
         orderBy('product_count',ascending=0)
         .drop('product_id').limit(5)

         )
display(most_df)
```

**Table**

| | product_name | product_count |
|---|---|---|
| 1 | sandwich | 48 |
| 2 | Chowmin | 24 |
| 3 | PIZZA | 21 |
| 4 | Dosa | 12 |
| 5 | Biryani | 6 |

5 rows

# Top ordered items

```
from pyspark.sql.functions import count

most_df=(sales_df.join(menu_df,'product_id').groupBy('product_id','product_name')
         .agg(count('product_id').alias('product_count')).
         orderBy('product_count',ascending=0)
         .drop('product_id').limit(1)

         )
display(most_df)
```

Table **Visualization 1**

# 48

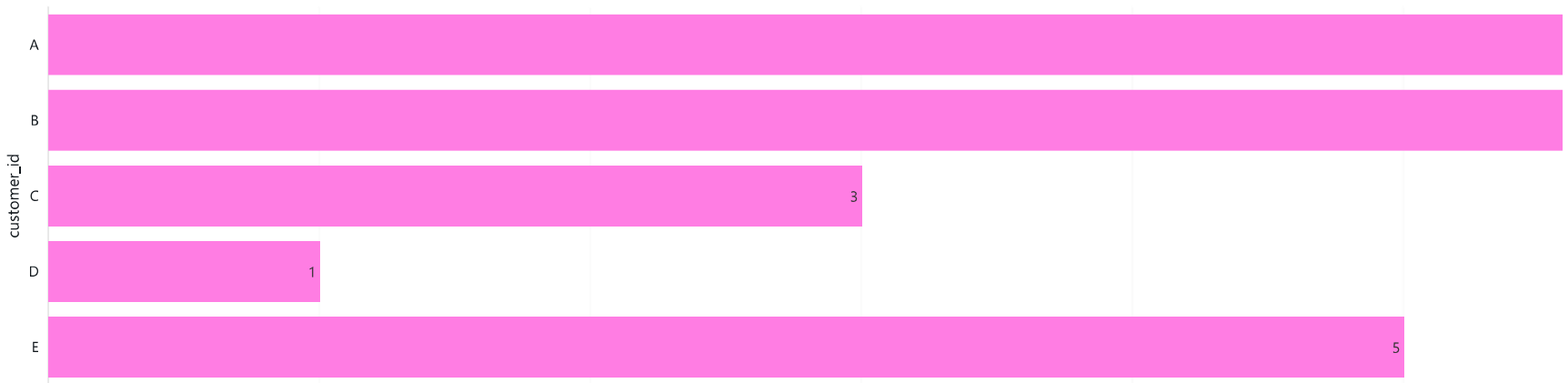
# ( sandwich)

1 row

## Frequency of customer visited

```
from pyspark.sql.functions import countDistinct
df =(sales_df.filter(sales_df.source_order=='Restaurant')
     .groupBy('customer_id')
     .agg(countDistinct('order_date'))
      )
display(df)
```

Table **Visualization 1**

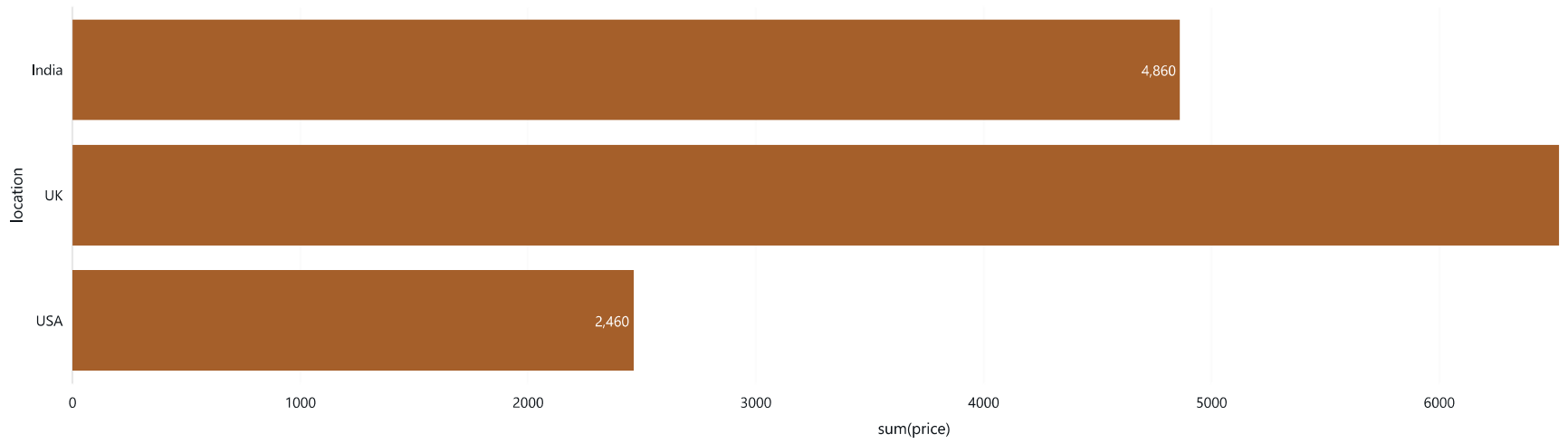



5 rows

## Total sales by each country

```
total_amount_spent = (sales_df.join(menu_df,'product_id').groupBy('location').agg({'price':'sum'}))
display(total_amount_spent)
```

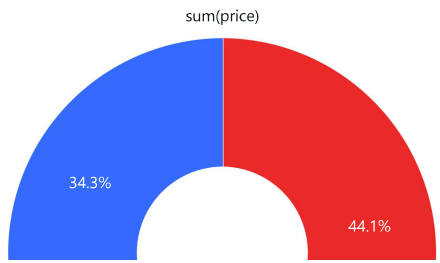
Table **Visualization 1**

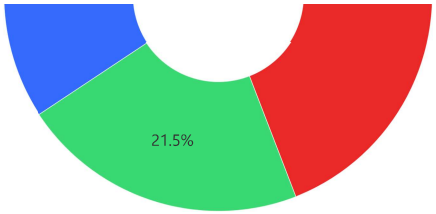



3 rows

## Total sales by order_source

Table **Visualization 1**






3 rows