Technical Challenge Solution

Oscar Garcia

To have a scalable solution that can solve this problem on machines without much RAM I used Pyspark instead of pandas. In the jupyter notebook I leave links with the instructions to install spark on a windows computer.

Data Cleaning

When all distinct states are printed we can see that there are 36 instead of 32. On the notebook I print an example for each of these. Two of these can be solved with the right options on the reading functions and are associated with the “ character. The other two bad states are filtered out. With this we get the proper 32 states.

```
df = spark.read.option("quote", "\"").option("escape", "\"").csv(path, header=True, inferSchema= True)
df = df.filter((col("estado") != "None") & (col("estado") != "estado"))
```

Q1. How many commercial chains are monitored?

This is calculated with the countDistinct() function on the cadenaCommercial column. There are 704 different chains.

```
    df.agg(countDistinct("cadenaComercial").alias("Number of commercial chains")).collect()[0]
```

```
Row(Number of commercial chains=704)
```

Q2. What are the top 10 monitored products by State?

To answer this questions I define a product as the combination of the product name, its presentation and brand. I define a new column that concatenates these three called “unique product” and use it to call a group by with state to use the count function to generate a new dataframe that have counted each unique product on each state.

```
df = df.withColumn("unique product", concat(col("producto"), lit(" "), col("presentacion"), lit(" "), col("marca")))
prod_counts = df.groupby("estado","unique product").count()
```

So that I can rank the products for each state I define a Window partitioned(or group) by state and ordered by their count in descending order. The row_number function will make use of the window to assign the proper rank (without considering ties) then I select only those rows with a rank of 10 or less.

```
window = (Window
        .partitionBy(prod_counts["estado"])
        .orderBy(prod_counts["count"].desc()))

res = (prod_counts.select(col("*"), row_number().over(window).alias("rank"))
.filter(col("rank") <= 10)
.orderBy(col("estado"), col("count").desc()))
```

To visualize the result I group by state and make a pivot table on their rank. This will result on one row for each state that shows their top 10 products. I show a small part of that table here.

```
res_pd = res.drop("count").groupby("estado").pivot("rank").agg(first("unique product")).orderBy(col("estado")).toPandas()
```

	estado	1	2	3	4	5	6
0	AGUASCALIENTES	TORTILLA DE MAIZ 1 KG. GRANEL S/M	LECHE ULTRAPASTEURIZADA CAJA 1 LT. ENTERA LALA...	LECHE PASTEURIZADA CAJA 1 LT. LALA. ENTERA	PAN BLANCO BOLILLO PIEZA S/M	BLANQUEADOR BOTELLA 950 ML. EL RENDIDOR CLORALEX	ACEITE BOTELLA 1 LT. MIXTO 1-2-3
1	BAJA CALIFORNIA	TORTILLA DE MAIZ 1 KG. GRANEL S/M	BLANQUEADOR BOTELLA 950 ML. EL RENDIDOR CLORALEX	ACEITE BOTELLA 1 LT. MIXTO 1-2-3	LECHE CONDENSADA LATA 397 GR. LA LECHERA.	REFRESCO LATA 355 ML. COCA COLA	ATUN LATA 140 GR. EN ACEITE. LOMO DOLORES

Q3. Which is the commercial chain with the highest number of monitored products?

I use the same definition of unique product as the last question. To answer I group by commercial chain and unique product and call a count function then order by it on descending order and take the first row. The commercial chain with the hidhest number of monitored products is Wal-Mart with 7371 unique products.

```
df.groupby("cadenaComercial").agg(countDistinct("unique product").alias("count")).orderBy(col("count").desc()).show(1)
```

```
+---------------+-----+
|cadenaComercial|count|
+---------------+-----+
|       WAL-MART| 7371|
+---------------+-----+
```

Q4. Use the data to find an interesting fact

We can see that in every state the 1 KG. tortilla is the top 1 product except on Distrito Federal in which it appears on second, this is to be expected as maize is the staple food on Mexico. On second place we tend to see various kinds of milk.These statistic count the appearance of the product on different stores, we can conclude that they are widespread but this does not tell us how many of these are being sold.

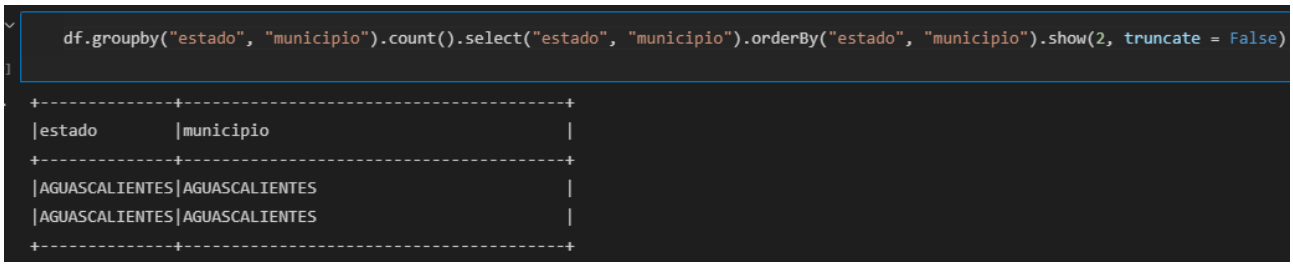
	estado	1
0	AGUASCALIENTES	TORTILLA DE MAIZ 1 KG. GRANEL S/M
1	BAJA CALIFORNIA	TORTILLA DE MAIZ 1 KG. GRANEL S/M
2	BAJA CALIFORNIA SUR	TORTILLA DE MAIZ 1 KG. GRANEL S/M
3	CAMPECHE	TORTILLA DE MAIZ 1 KG. GRANEL S/M

Q5. What are the lessons learned from this exercise?

We must ensure the data quality of the database before performing analysis. Failing to do this might invalidate an analysis and we might not catch this error before its too late.

Q6. Can you identify other ways to approach this problem? Explain.

If I had more time we would need to do analysis on each of the columns first to try and detect entry errors. One problem I identified is that there are city names that are entered incorrectly perhaps due to an extra white space.



```
df.groupby("estado", "municipio").count().select("estado", "municipio").orderBy("estado", "municipio").show(2, truncate = False)
```

```
+--------------+---------------------------------------+
|estado        |municipio                              |
+--------------+---------------------------------------+
|AGUASCALIENTES|AGUASCALIENTES                         |
|AGUASCALIENTES|AGUASCALIENTES                         |
+--------------+---------------------------------------+
```