

Universidad ORT Uruguay

Facultad de Ingeniería
Bernard Wand Polak

Herramientas de software para Big Data

Obligatorio

Facundo Aguerre – 187426

Gonzalo Nicolari – 262731

Gilmar Prates – 199374

Docentes

Alexis Arriola

Juan Rodriguez

Declaración de autoría

Nosotros, Facundo Aguerre, Gonzalo Nicolari y Gilmar Prates, declaramos que el trabajo que se presenta en esa obra es de nuestra propia mano. Podemos asegurar que:

- La obra fue producida en su totalidad mientras realizábamos el trabajo obligatorio de la materia Gestión de comunicación, conflictos en proyectos.
- Cuando hemos consultado el trabajo publicado por otros, lo hemos atribuido con claridad.
- Cuando hemos citado obras de otros, hemos indicado las fuentes. Con excepción de estas citas, la obra es enteramente nuestra.
- En la obra, hemos acusado recibo de las ayudas recibidas.
- Cuando la obra se basa en trabajo realizado conjuntamente con otros, hemos explicado claramente qué fue contribuido por otros.
- Ninguna parte de este trabajo ha sido publicada previamente a su entrega, excepto donde se han realizado las aclaraciones correspondientes.

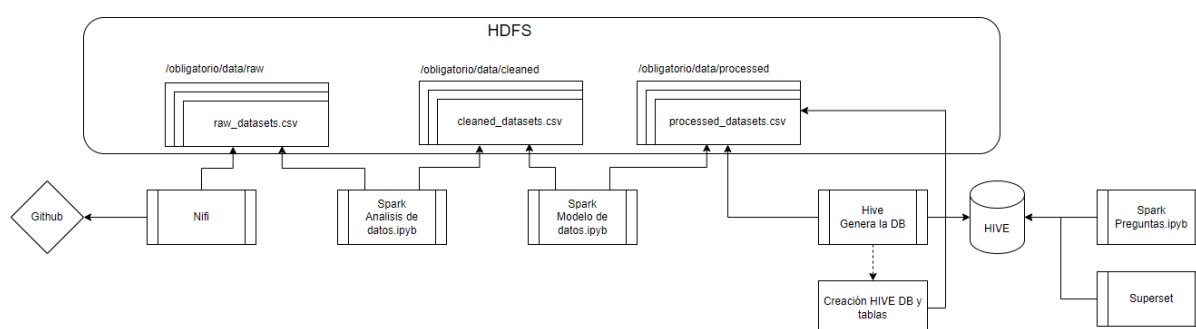
Índice

Parte 1.....	3
Arquitectura del data lake.....	3
Preparación de la ingesta de datos.....	3
Regiones definidas en HDFS para trabajar con los datos.....	4
Análisis exploratorio de los datos.....	4
Modelado de datos.....	5
Bosquejo del modelo.....	5
Descripción de las tablas.....	6
Guardado de dataframes en Hive.....	8
Resultados de preguntas seleccionadas a partir de tablas de Hive.....	9
Parte 2.....	19
Propuesta de data lake con tecnologías diferentes.....	19
Flujo completo.....	19
AWS Glue.....	19
AWS S3.....	20
AWS Redshift.....	21
AWS QuickSight.....	21
Bibliografía.....	23

Parte 1

Como fuente de datos, seleccionamos un dataset público de un E-Commerce de Brasil, el cual está referenciado en la bibliografía.

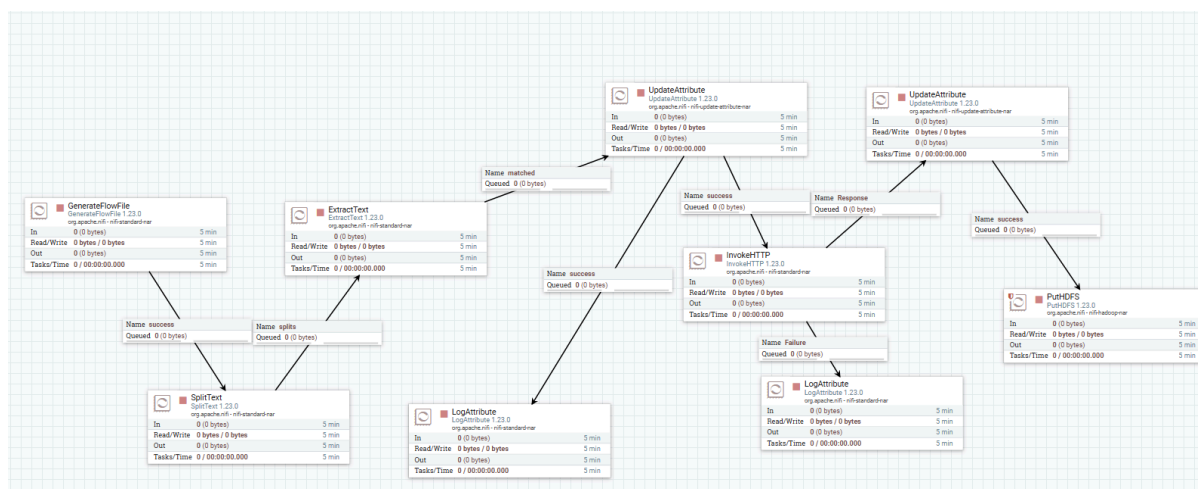
Arquitectura del data lake



Preparación de la ingesta de datos

En Nifi, primeramente generamos un GenerateFlowFile de forma que podamos concentrar todos los links referentes a la fuente de datos (GitHub: https://github.com/gpl9/BigData_Obligatorio). De esta forma, no es necesario generar un InvokeHTTP por cada una de las fuentes y podemos hacerlo recursivo.


A partir de esto, podemos realizar la transformación de cada una de las URLs y realizar un único InvokeHTTP como processor para realizar la llamada a la fuente de datos y obtener los recursos. Con UpdateAttribute actualizamos los nombres correspondientes a los archivos y finalmente, con el procesador PutHDFS, se alojan los datasets en la ruta establecida de HDFS (/obligatorio/data/raw).





Browse Directory


/obligatorio/data/raw

Go!



























Show

25

entries

Search:

<input type="checkbox"/>	 Permission	 Owner	 Group	 Size	 Last Modified	 Replication	 Block Size	 Name	
<input type="checkbox"/>	-rw-r--r--	ort	supergroup	8.62 MB	Dec 06 20:21	1	128 MB	olist_customers_dataset.csv	
<input type="checkbox"/>	-rw-r--r--	ort	supergroup	58.44 MB	Dec 06 20:21	1	128 MB	olist_geolocation_dataset.csv	
<input type="checkbox"/>	-rw-r--r--	ort	supergroup	14.72 MB	Dec 06 20:23	1	128 MB	olist_order_items_dataset.csv	
<input type="checkbox"/>	-rw-r--r--	ort	supergroup	5.51 MB	Dec 06 20:24	1	128 MB	olist_order_payments_dataset.csv	
<input type="checkbox"/>	-rw-r--r--	ort	supergroup	13.68 MB	Dec 06 20:24	1	128 MB	olist_order_reviews_dataset.csv	
<input type="checkbox"/>	-rw-r--r--	ort	supergroup	16.84 MB	Dec 06 20:25	1	128 MB	olist_orders_dataset.csv	
<input type="checkbox"/>	-rw-r--r--	ort	supergroup	2.27 MB	Dec 06 20:25	1	128 MB	olist_products_dataset.csv	
<input type="checkbox"/>	-rw-r--r--	ort	supergroup	170.61 KB	Dec 06 20:25	1	128 MB	olist_sellers_dataset.csv	
<input type="checkbox"/>	-rw-r--r--	ort	supergroup	2.48 KB	Dec 06 20:25	1	128 MB	product_category_name_translation.csv	

Showing 1 to 9 of 9 entries

Previous

1

Next

Regiones definidas en HDFS para trabajar con los datos

Ruta	Origen de creación	Propósito
/obligatorio/data/raw	NiFi	Alojar los datasets originales.
/obligatorio/data/cleaned	Jupyter notebook	Alojar los datasets refinados.
/obligatorio/data/processed	Jupyter notebook	Alojar los datos modelados.
/obligatorio/data/data_analytics	Hive	Alojar los datasets con los resultados de las preguntas 2 y 5.

Análisis exploratorio de los datos

Se realizó el análisis exploratorio de los datos dentro de un Jupyter notebook que estará incluido en el comprimido de la entrega y también en el repositorio de GitHub (**Analisis Exploratorio.ipynb**).

Para el análisis, siendo que la limpieza de datos y control de valores es similar para todos los componentes, procedimos a realizar un análisis mediante un método que aplica a cada dataset el mismo análisis.

Casos donde se requería un trabajo extra individual, puede ser notado en el notebook.

Dentro de HDFS, se guardaron los nuevos datasets con los datos refinados, en la ruta: **/obligatorio/data/cleaned**

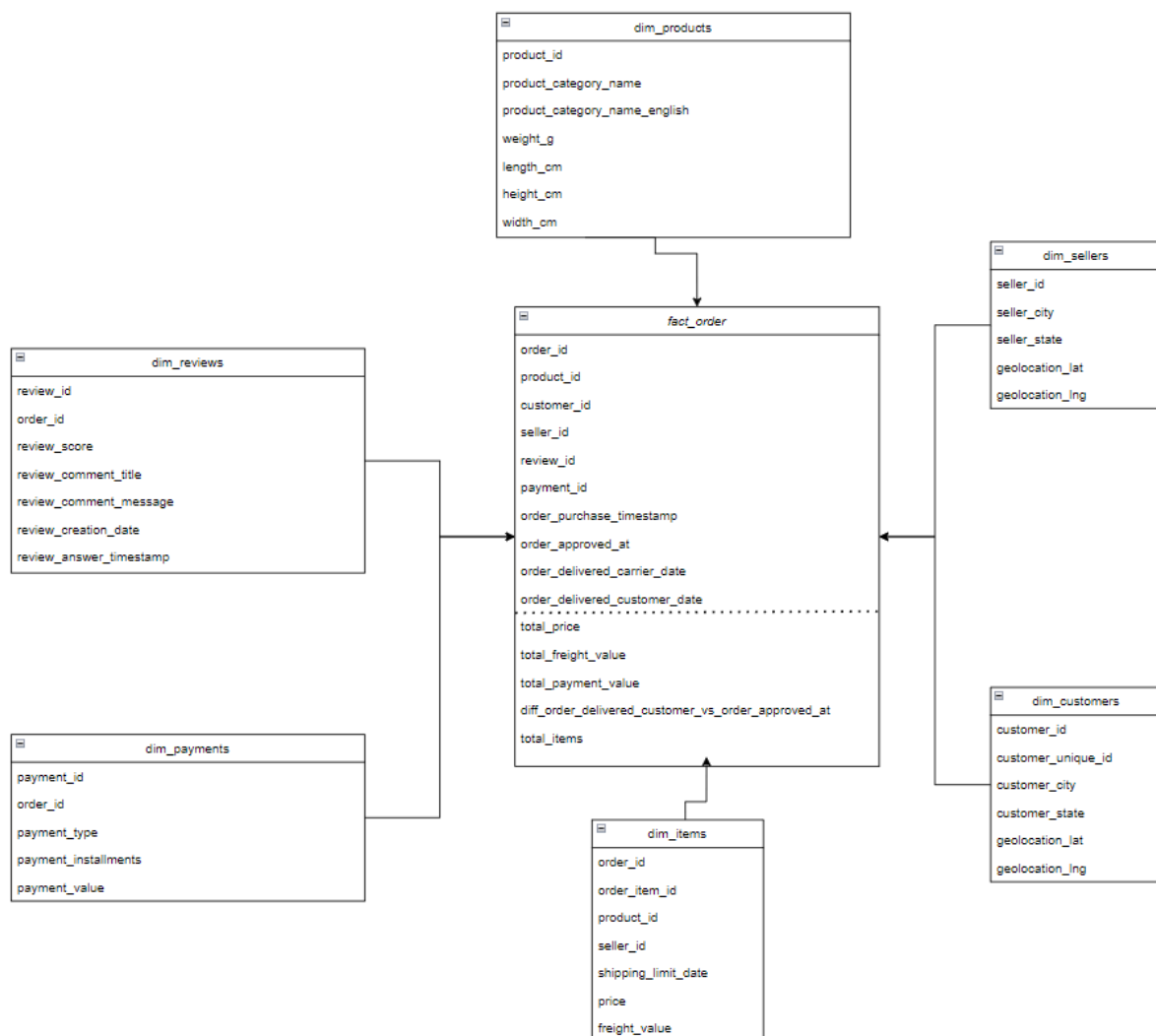
Modelado de datos

Elegimos modelar los datos con un **diagrama estrella** debido a su capacidad para equilibrar simplicidad, rendimiento y funcionalidad en el análisis de datos. Presentando una estructura clara y fácil de entender, con una tabla central que almacena los datos, lo que facilita la navegación y la comprensión.

Los datos se modelaron dentro de un Jupyter notebook que estará incluido en el comprimido de la entrega y también en el repositorio de GitHub (**Modelo De Datos.ipynb**).

Dentro de HDFS, se guardaron los nuevos datasets, en la ruta: **/obligatorio/data/processed**

Bosquejo del modelo



Descripción de las tablas

- dim_customers
 - customer_id - String
 - Se refiere al id único del customer. Puede coincidir con un id de seller.
 - customer_unique_id - String
 - Se refiere al id único del customer.
 - customer_city - String
 - Se refiere a la ciudad donde pertenece el customer.
 - customer_state - String
 - Se refiere al estado donde se encuentra la ciudad del customer.
 - geolocation_lat - String
 - Punto de latitud donde se ubica el customer.
 - geolocation_lng - String
 - Punto de longitud donde se ubica el customer.
- dim_items
 - order_id - String
 - Identificador de la orden.
 - order_item_id - Int
 - Identificador del order item.
 - product_id - String
 - Identificador del producto.
 - seller_id - String
 - Identificador del seller.
 - shopping_limit_date - String
 - Fecha límite de venta.
 - price - Double
 - Precio de venta.
 - freight_value - Double
 - Valor del envío.
- dim_payments
 - payment_id - BigInt
 - Identificador único de pago.
 - order_id - String
 - Identificador único de la orden.
 - payment_type - String
 - Tipo de pago asociado.
 - payment_installments - Int
 - Cantidad de pagos.
 - payment_value - Double

- Valor del pago.
- dim_products
 - product_id - String
 - Identificador único del producto.
 - product_category_name - String
 - Nombre de categoría del producto.
 - product_category_name_english - String
 - Nombre de la categoría del producto en inglés.
 - weight_g - Int
 - Medida Peso del producto.
 - length_cm - Int
 - Medida Largo del producto.
 - height_cm - Int
 - Medida Altura del producto.
 - width_cm - Int
 - Medida Ancho del producto.
 -
- dim_reviews
 - review_id - String
 - Identificador único de la reseña.
 - order_id - String
 - Identificador único de la orden.
 - review_score - Int
 - Valor de la reseña [1-5].
 - review_comment_title - String
 - Título asociado a la reseña.
 - review_comment_message - String
 - Comentario asociado a la reseña.
 - review_creation_date - String
 - Fecha de creación de la reseña.
 - review_answer_timestamp - String
 - Fecha de respuesta de la reseña.
- dim_sellers
 - seller_id - String
 - Identificador único del vendedor.
 - seller_city - String
 - Ciudad asociada al vendedor.
 - seller_state - String
 - Estado asociado del vendedor.
 - geolocation_lat - String
 - Punto de latitud donde se ubica el vendedor.
 - geolocation_lng - String

- Punto de longitud donde se ubica el vendedor.
- fact_order
 - order_id - String
 - Identificador de la orden.
 - product_id - String
 - Identificador del producto.
 - customer_id
 - Identificador del customer.
 - seller_id - String
 - Identificador del seller.
 - payment_id - BigInt
 - Identificador del pago.
 - order_purchase_timestamp - String
 - Fecha de la orden
 - order_approved_at - String
 - Fecha de aprobación de la orden.
 - order_delivered_carrier_date - String
 - Fecha de envío de la orden.
 - order_delivered_customer_date - String
 - Fecha de entrega de la orden al customer.
 - total_price - Double
 - Precio total.
 - total_freight_value - Double
 - Valor total del envío.
 - total_payment_value - Double
 - Valor total pago.
 - diff_order_delivered_customer_vs_order_approved_at - Double
 - Valor referente al tiempo desde que sucedió la entrega y fue aprobada para realizarse. Se refiere al tiempo que se llevó entre que se aprobó la orden y el cliente la recibió.
 - total_items - BigInt
 - Total de items.

Guardado de dataframes en Hive

Se creó la base de datos en Hive con el nombre **obligatorio** y sobre la cual creamos las tablas para cada dataset almacenado en HDFS, ruta: **/obligatorio/data/processed**

El código para crear la base de datos y sus respectivas tablas, estará incluido en el comprimido de la entrega y también en el repositorio de GitHub (**Creación HIVE DB y tablas.txt**).

Resultados de preguntas seleccionadas a partir de tablas de Hive

Las preguntas que seleccionamos para realizar su investigación, fueron:

1. Top 10 conjuntos de productos que más se venden juntos.

A nivel organización y Olist, siendo una de los mayores e-commerce de Brasil, entendemos que la venta de productos conjuntos es una información muy valiosa para los distintos departamentos. Por ejemplo, el departamento de marketing puede verse beneficiado de esta información dado que podría permitirle establecer por ciudad posibles publicidades o promociones.

```
set_products = spark.sql("""
    SELECT fo.product_id AS product_1, fo2.product_id AS product_2, COUNT(*) AS pair_count
    FROM fact_order fo
    JOIN fact_order fo2 ON fo2.order_id = fo.order_id AND fo2.product_id < fo.product_id
    GROUP BY fo.product_id, fo2.product_id
    ORDER BY pair_count DESC
    LIMIT 10
""")
```

```
set_products.show(truncate=False)
```

```
[Stage 1:>] (0 + 2) / 2]
+-----+-----+-----+
|product_1|product_2|pair_count|
+-----+-----+-----+
|ebf9bc6cd60eadd681384e3116fda85|5ddab10d5e0a23acb99acf56b62b3276|441|
|d1c427060a0f73f6b889a5c7c61f2ac4|2b939dc9b176d7fa21594d588815d4a4|144|
|ebf7d9b8166c02eb72e573eebd630458|18796df281656da4036dd926561a6030|100|
|4754d89182db4010eabfb20da5fb7191|24aba57735be13fd785bc04d1a8812e4|36|
|9c7bdf67b06b419aefb93cfdcf96c55d|45e9db074ca64dc81ee6f06185544d6e|36|
|85d4c1a46f08f730de651ea6f6645313|19944bc70a13c9b62e18a2bc7e913bb6|36|
|99a4788cb24856965c36a24e339b6058|35afc973633aaeb6b877ff57b2793310|35|
|e53e557d5a159f5aa2c5e995d9df244b|36f60d45225e60c7da4558b070ce4b60|34|
|dd768d259ee6054e0dadd66c8e2be0b6|0a108ad81c4e2a790649c3f8070f1213|25|
|f2c1df0fc307ec60e3321c708c1dcac3|99a4788cb24856965c36a24e339b6058|25|
+-----+-----+-----+
```

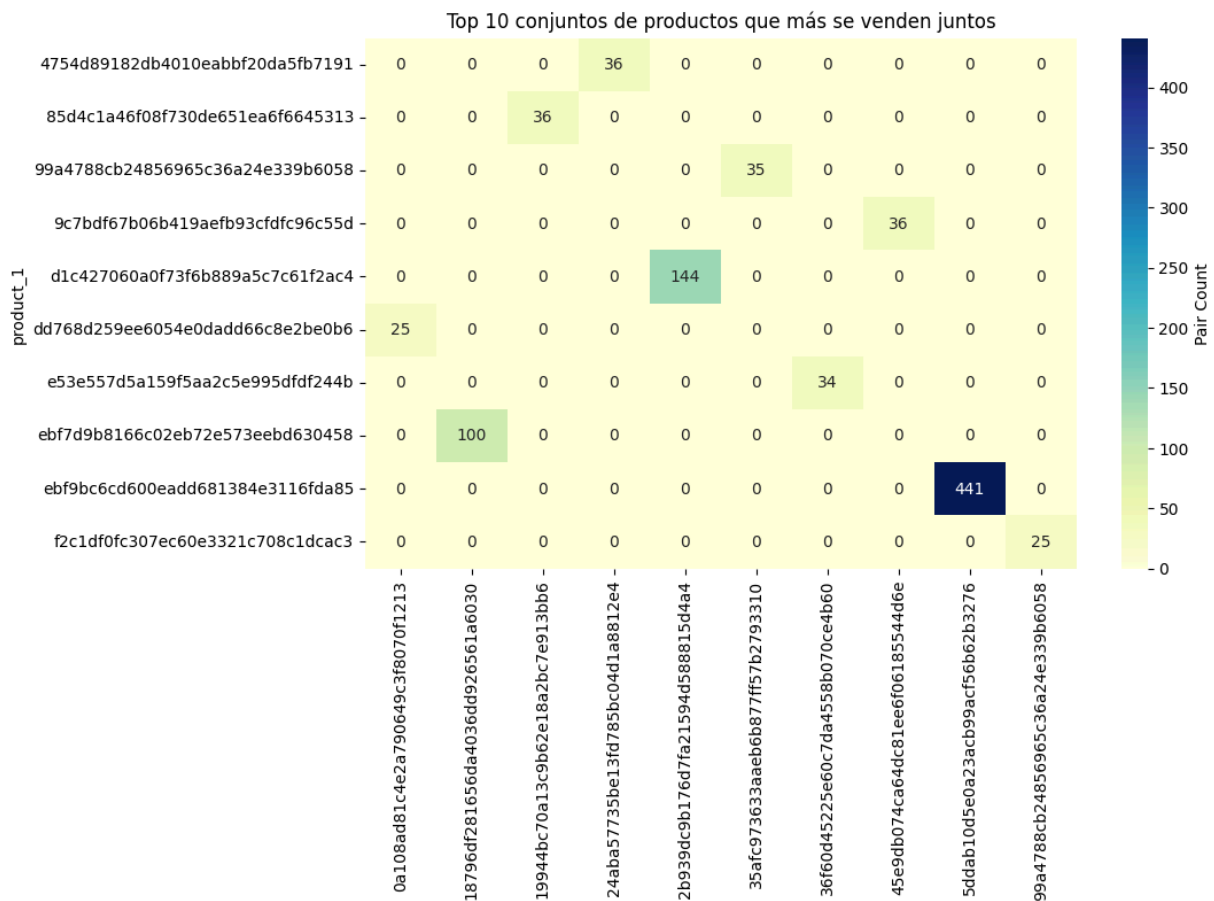
Representamos los resultados mediante dos tipos de gráficas en la notebook de Jupyter:

Mapa de calor

```
set_products_df = set_products.toPandas()
```

```
heatmap_data = set_products_df.pivot_table(index='product_1', columns='product_2', values='pair_count', fill_value=0)
```

```
plt.figure(figsize=(10, 6))
sns.heatmap(heatmap_data, annot=True, cmap='YlGnBu', fmt='g', cbar_kws={'label': 'Pair Count'})
plt.title('Top 10 conjuntos de productos que más se venden juntos')
plt.show()
```

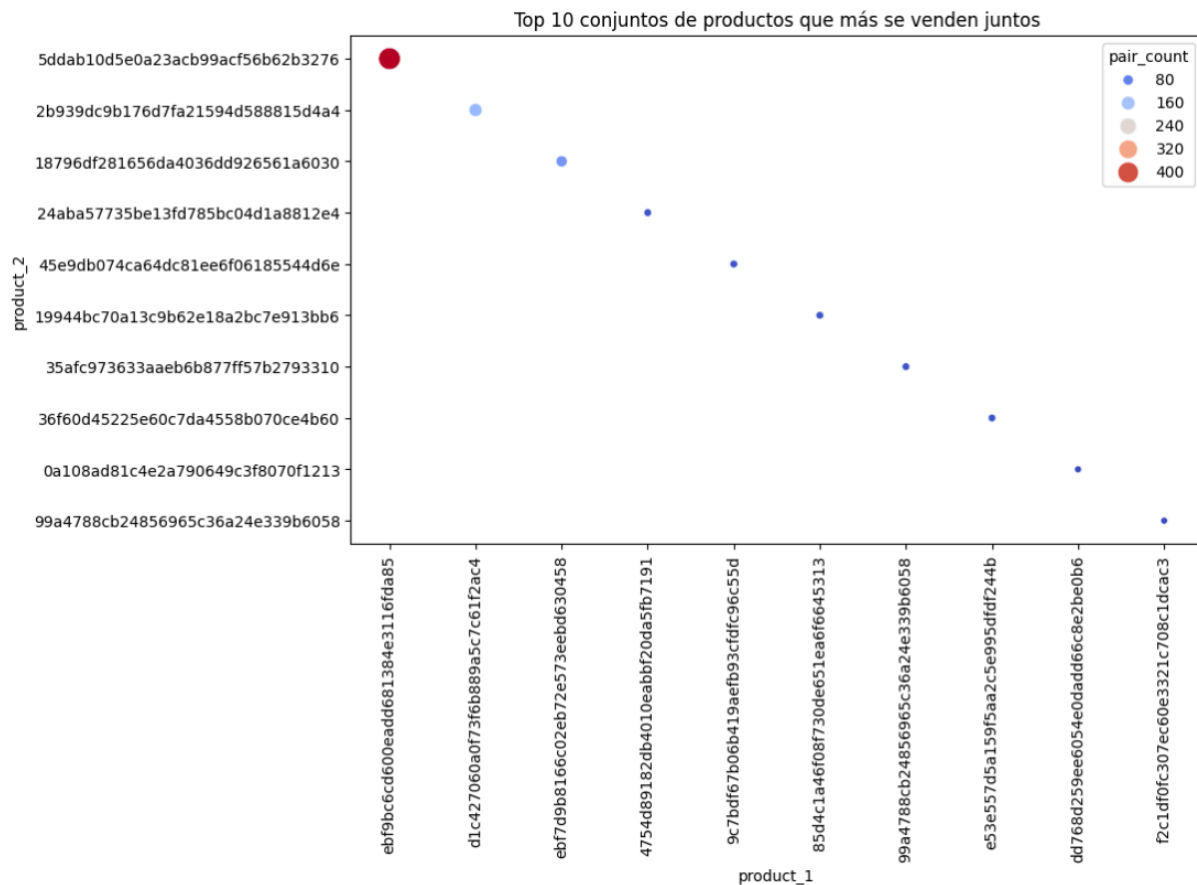


Para el caso del mapa de calor, podemos identificar qué productos conjuntos (basado en los ejes de las xy) son los que mayor cantidad de ventas tienen en conjunto.

Scatter Plot

```
set_products_df = set_products.toPandas()

plt.figure(figsize=(10, 6))
sns.scatterplot(data=set_products_df, x='product_1', y='product_2', size='pair_count', hue='pair_count', palette='coolwarm', sizes=(20, 200))
plt.xticks(rotation=90)
plt.title('Top 10 conjuntos de productos que más se venden juntos')
plt.show()
```



De igual manera que lo anterior, representando en ejes xy se identifica con mayor tamaño y color los valores de productos más vendidos.

2. Top 5 productos más vendidos por ciudad.

Para el caso de los productos más vendidos por ciudad, se mantiene la búsqueda de información relacionada a los productos de forma que permita a las distintas áreas del negocio, obtener datos en los cuales puedan basar sus decisiones.

Por ejemplo, el equipo de logística de una ciudad respectiva, podría utilizar esta información para el manejo de stock de productos críticos de ventas.

```
ranked_products = spark.sql("""
    WITH ranked_products AS (
        SELECT dc.customer_city AS city, fo.product_id, COUNT(fo.product_id) AS product_count,
               ROW_NUMBER() OVER (PARTITION BY dc.customer_city ORDER BY COUNT(fo.product_id) DESC) AS rank
        FROM fact_order fo
        JOIN dim_customers dc ON dc.customer_id = fo.customer_id
        GROUP BY dc.customer_city, fo.product_id
    )
    SELECT city, product_id, product_count, rank
    FROM ranked_products
    WHERE rank <= 5
    ORDER BY city, rank;
""")

ranked_products.show(500, truncate=False)
```

city	product_id	product_count	rank
abadia dos dourados	418d480693f2f01e9cf4568db0346d28	21	1
abadia dos dourados	c1aabb6f4caec9f5bf7cd80519d6cc0	21	2
abadia dos dourados	1081ae52311daac87fb54ba8ce4670ac	21	3
abadiania	0a9b9a871ffaec6c0198334558a6c6a1	23	1
abaete	3354a4e684f5e7199f9407db70ccd92b	109	1
abaete	b84a91f1e58b1f44daf3a9b74a83f1d8	109	2
abaete	5d136eed606105a39f8cffd4b7978951	109	3
abaete	97170c28efde1b426f1c52348fe1c086	109	4
abaete	e0d64dcfaa3b6db5c54ca298ae101d05	109	5
abaetetuba	575889c312f07612d39648ea61253cd5	250	1
abaetetuba	008cff0e5792219fae03e570f980b330	125	2
abaetetuba	1332f9dadde087afb818a88c72db05f1	125	3
abaetetuba	78819ab10b4a44927c832a21c06849e0	125	4
abaetetuba	8a0234d9460e4ad9a041849057dce64e	125	5
abaiara	9c60e93e03989cf6e4a7da214723920d	4	1
abaiara	629e019a6f298a83aecc7877964f935	4	2
abaira	ca0ebb053b17d7de027780de787ed000	11	1
abaira	91b08d34d0ba4db44da2dc382867ba49	11	2
abare	a367f823feb49954fedb4f4a43ad9db2	21	1
abare	543afaf83c3e39a4c4a6df503b65ff58	21	2
abatia	35afc973633aaeb6b877ff57b2793310	22	1
abatia	59ebd2c07d59483d0855b5b3d5e18728	22	2
abatia	87e583842cc9122ded9f8eafb42894c	22	3

Representamos los resultados mediante una gráfica en Superset:

Tabla

city	product_id	product_count	rank
abadia dos dourados	418d480693f2f01e9cf4568db0346d28	21	1
abadia dos dourados	1081ae52311daac87fb54ba8ce4670ac	21	2
abadia dos dourados	c1aabb6f4caec9f5bf7cd80519d6cc0	21	3
abadiania	0a9b9a871ffaec6c0198334558a6c6a1	23	1
abaete	5d136eed606105a39f8cffd4b7978951	109	1
abaete	b84a91f1e58b1f44daf3a9b74a83f1d8	109	2
abaete	cdce55fb5f6e6b24f00f27364a8afde3	109	3
abaete	3354a4e684f5e7199f9407db70ccd92b	109	4
abaete	9ad75bd7267e5c724cb42c71ac56ca72	109	5
abaetetuba	575889c312f07612d39648ea61253cd5	250	1
abaetetuba	c8a4ec6d4201d92e157eafb89eab932b	125	2
abaetetuba	a7c87b1bbdd51e0d88b0307cfd03d47	125	3
abaetetuba	06c6e01186af8b98ee1fc9e01f9471e9	125	4
abaetetuba	1332f9dadde087afb818a88c72db05f1	125	5
abaiara	9c60e93e03989cf6e4a7da214723920d	4	1
abaiara	629e019a6f298a83aecc7877964f935	4	2
abaira	91b08d34d0ba4db44da2dc382867ba49	11	1
abaira	ca0ebb053b17d7de027780de787ed000	11	2
abare	a367f823feb49954fedb4f4a43ad9db2	21	1
abare	543afaf83c3e39a4c4a6df503b65ff58	21	2
abatia	87e583842cc9122ded9f8eafb42894c	22	1
abatia	35afc973633aaeb6b877ff57b2793310	22	2
abatia	59ebd2c07d59483d0855b5b3d5e18728	22	3

Encontramos como más favorable la representación de una tabla que nos indique, por ciudad, el top de productos más vendidos dejando en claro también, la cantidad de estas ventas.

3. ¿Qué días del mes se vende más?

Importante métrica que puede ser valorada considerablemente por el equipo de marketing de forma de entender cuando realizar campañas publicitarias o promociones.

Combinado con la pregunta anterior, puede dar información relevante sobre la comprensión de los clientes.

```
fact_order = spark.table("fact_order")
```

```
days_sales = fact_order.select(
    dayofmonth(col("order_purchase_timestamp")).alias("day_of_month"),
    col("total_items")
).groupBy("day_of_month").agg(
    sum("total_items").alias("total_products_sold")
)
```

```
most_sold_day = days_sales.filter(col("total_products_sold").isNotNull()) \
    .orderBy(col("total_products_sold").desc())
```

```
most_sold_day.show(31)
```

day_of_month	total_products_sold
24	50706214888
16	46849996228
15	45515622152
4	44891281988
5	44781104312
18	44560748960
6	44266941824
19	43703811480
7	43581391840
8	43397762380
26	43312068632
14	43312068632
11	42908083820
25	42614276684
20	42406163296
13	42247017764
3	42198049908
17	41989936520
9	41867516880
2	41769581168
12	41475774032
22	41132999040
27	40875917796
1	40569868696
23	40484174948
10	40471932984
21	40386239236
28	38831509808
29	33445045648
30	32600350132
31	21252049504

Representamos los resultados mediante dos tipos de gráficas en la notebook de Jupyter:

Gráfico de barras

```
most_sold_day_pd = most_sold_day.toPandas()

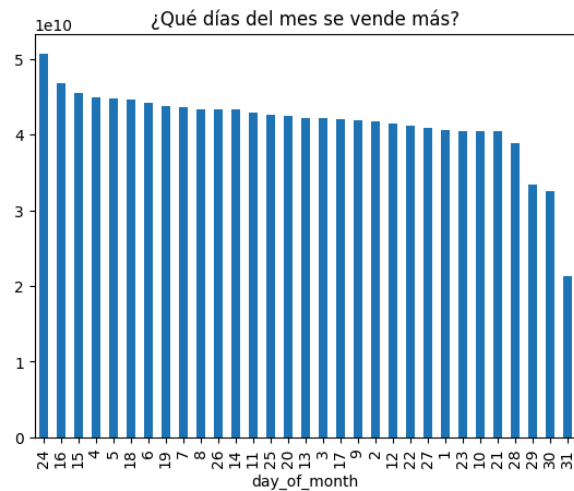
pandas_bokeh.output_notebook()

most_sold_day_pd.plot(kind='bar', x='day_of_month', y='total_products_sold', title='¿Qué días del mes se vende más?', legend=False)
```



BokehJS 3.1.1 successfully loaded.

<Axes: title={'center': '¿Qué días del mes se vende más?'}, xlabel='day_of_month'>



Visualmente, el gráfico de barras aporta una forma rápida y clara de identificar cuál es el día que más se vende conjuntamente con la comparativa de los días que le siguen.

Diagrama de líneas

```
sns.set(style="whitegrid")

plt.figure(figsize=(12, 6))
sns.lineplot(
    x="day_of_month",
    y="total_products_sold",
    data=most_sold_day_pd,
    marker="o",
    color="b"
)

plt.xticks(most_sold_day_pd["day_of_month"], rotation=45)
plt.xlabel("Día del mes", fontsize=12)
plt.ylabel("Total ventas", fontsize=12)
plt.title("¿Qué días del mes se vende más?", fontsize=16)

plt.grid(color='gray', linestyle='--', linewidth=0.5)

plt.show()
```



El diagrama de líneas, al igual que el gráfico anterior, permite hacer la cronología de los días y sus respectivas ventas. Permite de manera visual, hacer un control no solo de los días que más se venden, sino la performance de las ventas a lo largo del mes.

4. ¿Qué proporción de pedidos recibe cada calificación en las reseñas (de 1 a 5)?

Permite conocer las calificaciones por parte de los clientes ayudando a visualizar si la mayoría de los clientes están satisfechos o no con el servicio ofrecido.

Ayuda también a identificar la tendencia general del cliente con la empresa. Es un termómetro para la empresa.

```
total_reviews = spark.sql("""
    SELECT COUNT(*) AS total_reviews
    FROM fact_order fo
    JOIN dim_reviews dr ON dr.order_id = fo.order_id
    """).collect()[0][0]
```

```
review_counts = spark.sql("""
    SELECT review_score, COUNT(*) AS review_count
    FROM fact_order fo
    JOIN dim_reviews dr ON dr.order_id = fo.order_id
    GROUP BY review_score
    """)
```

```
review_percentages = review_counts.withColumn("percentage", round((col("review_count") / total_reviews) * 100, 2))
```

```
review_percentages.show()
```

```
2024-12-11T21:46:27,657 WARN [Executor task launch worker for task 1.0 in stage 32.0 (TID 41)] org.apache.hadoop.hive.serde2.lazy.LazyStruct - Extra byte
s detected at the end of the row! Ignoring similar problems.
2024-12-11T21:46:27,660 WARN [Executor task launch worker for task 0.0 in stage 32.0 (TID 40)] org.apache.hadoop.hive.serde2.lazy.LazyStruct - Extra byte
s detected at the end of the row! Ignoring similar problems.
+-----+-----+
|review_score|review_count|percentage|
+-----+-----+
|      null|         4|         0.0|
|         1|       10060|         9.9|
|         3|        8404|         8.27|
|         5|       59941|       58.97|
|         4|       20035|       19.71|
|         2|        3203|         3.15|
+-----+-----+
```


Representamos los resultados mediante dos tipos de gráficas en la notebook de Jupyter:

Gráfico circular

```
review_percentages_pd = review_percentages.toPandas()

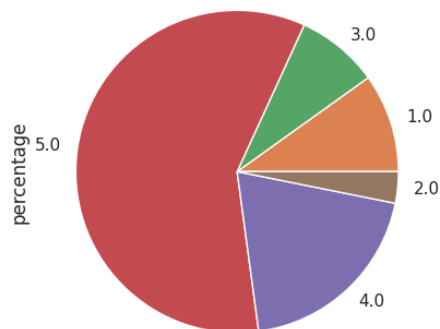
pandas_bokeh.output_notebook()

review_percentages_pd.plot(kind="pie", y="percentage", labels=review_percentages_pd["review_score"], legend=False, title="¿Qué proporción de pedidos reci
```

BokehJS 3.1.1 successfully loaded.

<Axes: title={'center': '¿Qué proporción de pedidos recibe cada calificación en las reseñas (de 1 a 5)?'}, ylabel='percentage'>

¿Qué proporción de pedidos recibe cada calificación en las reseñas (de 1 a 5)?



Para el caso del gráfico, es fácil ver, conjuntamente con la etiqueta, las calificaciones que predominan. A su vez, da una vista equilibrada al manejar proporciones de cuál es el mayor valor predominante.

Gráfico de barras

```

review_scores = review_percentages_pd['review_score']
percentages = review_percentages_pd['percentage']

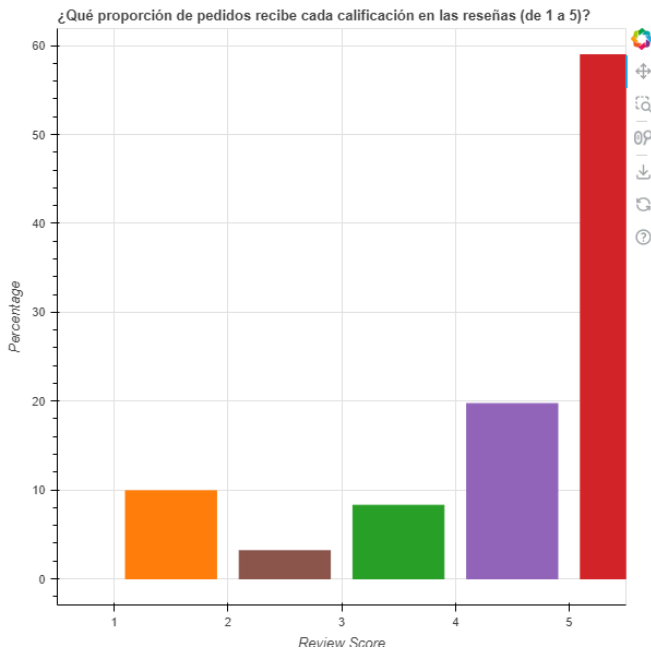
p = figure(title="¿Qué proporción de pedidos recibe cada calificación en las reseñas (de 1 a 5)?", x_axis_label='Review Score', y_axis_label='Percentage')

p.vbar(x=review_scores, top=percentages, width=0.8, color=colors[len(review_scores)])

output_notebook()
show(p)

```

BokehJS 3.1.1 successfully loaded.



Fácil de identificar y percibir los valores predominantes de las reseñas, siendo claramente posible ver los porcentajes que ocupan cada una de los 5 valores posibles.

5. ¿Cuál es el tiempo promedio entre la aprobación de un pedido y la entrega al cliente?

Para responder esta pregunta, buscamos un gráfico simple pero atractivo, y optamos por el velocímetro adaptado a segundos para ver el promedio y poder compararlo con los valores máximos y mínimos de entrega. Esto es de gran valor para ponerse objetivos alcanzables y medibles a nivel de empresa a fin de reducir tiempos de entrega.

```
fact_order = spark.table("fact_order")
```

```

avg_time = fact_order.agg(
    count("*").alias("total_orders"),
    round(avg("diff_order_delivered_customer_vs_order_approved_at"), 2).alias("average_time (days)")
)

```

```
avg_time.show()
```

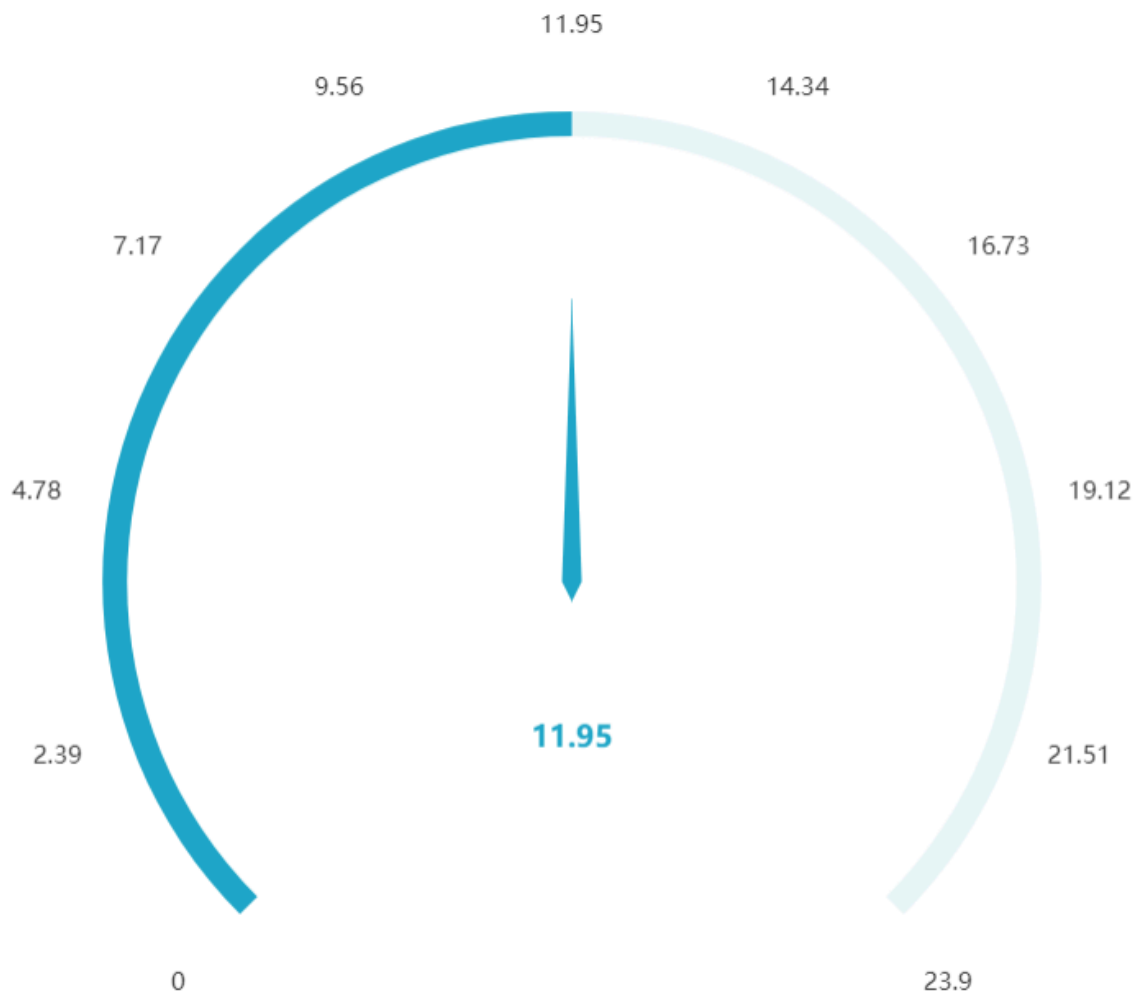
```

+-----+-----+
|total_orders|average_time (days)|
+-----+-----+
|      105247|             11.95|
+-----+-----+

```

Representamos los resultados mediante una gráfica en Superset:

Tabla de calibres



Parte 2

Propuesta de data lake con tecnologías diferentes

Se realizó una investigación para crear un data lake similar al de la parte 1, utilizando otras tecnologías. Se decidió ir por el lado de las herramientas de Amazon, por lo que se desarrolló el mismo flujo de datos pero con distintas herramientas, siendo las siguientes:

- Amazon Glue
- Amazon S3
- Amazon Redshift
- Amazon QuickSight

Flujo completo

Se trata de un proceso de ETL donde estaríamos obteniendo los datos desde una fuente, procediendo a realizar la transformación de los mismos (análisis, limpieza, pre-cálculos, estructuración) para luego realizar la carga de los datos al data lake y así, proceder desde un dominio de aplicación, realizar la consulta e interpretación de los mismos.

Pasos

- Ingesta: AWS Glue
- Almacenamiento datos raw: AWS S3
- Limpieza y transformación de datos: AWS Glue
- Almacenamiento de datos limpios y transformados: AWS S3
- Carga de datos a Redshift: AWS Redshift COPY
- Modelado y consultas: AWS Redshift
- Visualización: AWS QuickSight

AWS Glue

Seleccionamos Amazon Glue para la ingesta y transformación de los datos, debido a diferentes razones:

Amazon Glue cuenta con la herramienta de **Glue Crawlers**, es una herramienta que detecta automáticamente cambios en los datos almacenados en fuentes, y ejecuta scripts para catalogarlos y crear esquemas estructurados, eliminando la necesidad de configuraciones manuales.

AWS Glue también cuenta con scripts automáticos de ETL que Glue genera en Python o Scala basados en Apache Spark, reduciendo el esfuerzo de codificación.

Comparamos esta herramienta con Apache Nifi, la utilizada para realizar la parte 1, y sacamos las siguientes conclusiones:

Característica	Amazon Glue	Apache NiFi
Arquitectura	Serverless	Se requiere configuración manual
Automatización	Crawlers y generación de scripts	Configuración manual de flujos
Escalabilidad	Automática	Limitada a infraestructura local
Costos	Pago por uso	Costo fijo por infraestructura
Integración con AWS	Total	Parcial

AWS S3

Seleccionamos Amazon S3 como solución de almacenamiento para construir el data lake debido a sus características avanzadas, flexibilidad, compatibilidad y bajo costo con otras herramientas de AWS.

Sus puntos principales son la escalabilidad, velocidad, flexibilidad y seguridad. Es altamente escalable debido a que puede almacenar desde gigabytes hasta petabytes de datos sin necesidad de ajustes manuales, y todo esto acompañado de la garantía de acceso continuo a los datos debido a su distribución.

Aporta a la flexibilidad debido a la facilidad de organizar datos en zonas raw, procesado y listo para analítica, utilizando carpetas lógicas en vez de buckets.

Su seguridad es muy robusta, se pueden configurar permisos específicos a nivel de bucket o archivo utilizando distintas políticas.

Otro punto a favor al ser una herramienta nativa de Amazon, es que los datos pueden almacenarse en particiones lógicas, lo que mejora el rendimiento de consultas con herramientas como Amazon Redshift. El ser una herramienta nativa permite integrarse con Redshift pero también con Amazon Glue, lo que simplifica el flujo de trabajo desde la ingesta de datos hasta la obtención de insights analíticos.

AWS Redshift

Para el análisis de datos elegimos Amazon Redshift, que es un servicio de almacenamiento y análisis de datos diseñado para manejar grandes volúmenes de datos estructurados y semiestructurados con alto rendimiento.

Funciona como un data warehouse moderno, escalable y totalmente administrado, lo que permite ejecutar consultas complejas y análisis avanzado sin necesidad de administrar infraestructura física.

Además, al ser una herramienta del ecosistema de Amazon, se integra con otras herramientas nativas de AWS, como Amazon S3, AWS Glue y AWS QuickSight, lo que facilita un flujo de trabajo desde la ingesta hasta la visualización de datos.

Elegimos AWS Redshift también por su compatibilidad con formatos modernos como Parquet y ORC, que sumado a su capacidad para trabajar directamente con datos en Amazon S3, asegura una transición eficiente entre las etapas de limpieza, modelado y análisis, lo que facilita mucho el flujo de trabajo.

AWS QuickSight

Amazon QuickSight es una herramienta de visualización e inteligencia empresarial completamente administrada y diseñada para transformar datos en insights interactivos.

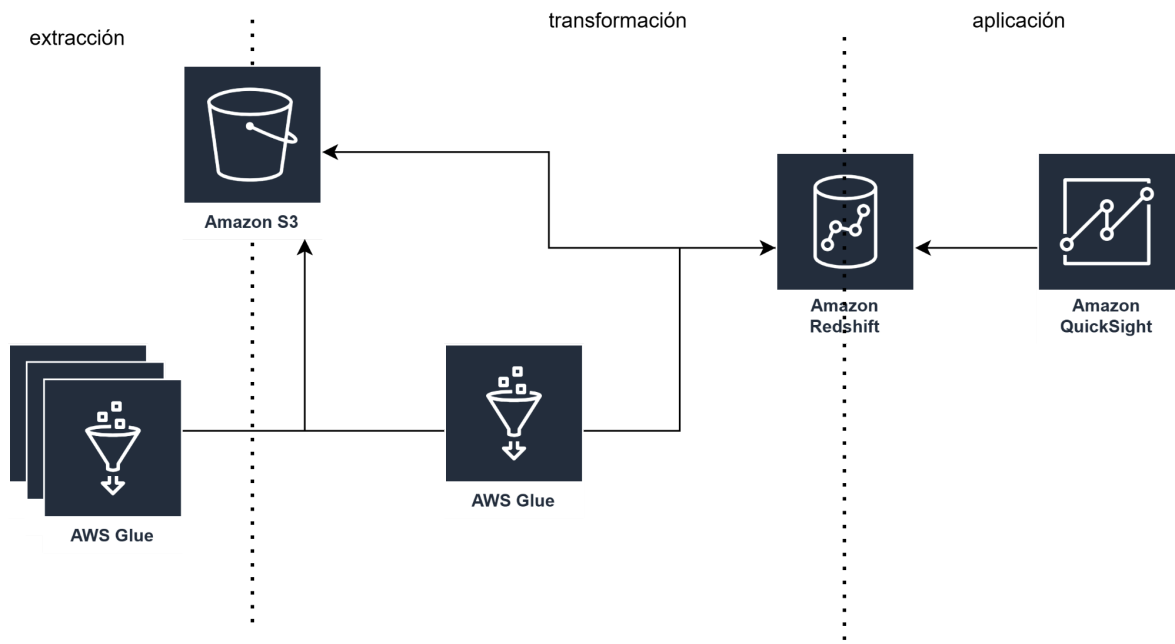
La elegimos debido a varias razones, una de ellas es que funciona como un servicio que permite crear dashboards dinámicos, gráficos y reportes visuales a partir de diversas fuentes de datos con una gran velocidad debido a su motor de búsqueda llamado SPICE (Super-fast, Parallel, In-memory Calculation Engine), que funciona mediante el almacenamiento de datos en memoria con almacenamiento columnar para mayor acceso y velocidad de análisis.

Al ser una herramienta nativa de AWS, facilita la integración con las otras herramientas a utilizar en el data lake.

Otra razón por la cual nos volcamos hacia QuickSight, es ofrece actualizaciones dinámicas de los dashboards al conectarse con fuentes de datos en vivo.

Su automatización y customización de insights utilizando machine learning, contribuye a que cualquier usuario sin necesidad de conocimiento del entorno pueda crear reportes, tendencias, etc. en base a los datos almacenados.

En términos de seguridad, es una herramienta muy potente, ya que utiliza distintas políticas para acceso, como AWS Identity and Access Management (IAM), OpenID, AWS Directory y demás. También utiliza permisos muy granulares de acceso, hasta seguridad a nivel de récord. Todo esto potenciado por una alta seguridad de encriptación.



Bibliografía

Fuente de datos E-Commerce de Brasil

<https://www.kaggle.com/datasets/olistbr/brazilian-ecommerce>

Amazon Glue

<https://docs.aws.amazon.com/glue/latest/dg/what-is-glue.html>

Amazon S3

<https://docs.aws.amazon.com/AmazonS3/latest/userguide/Welcome.html>

Amazon Redshift

<https://aws.amazon.com/es/redshift/>

Amazon QuickSight

<https://docs.aws.amazon.com/quicksight/latest/user/welcome.html>

Best Practices with Amazon Quick Sight

<https://rajaswalavalkar.medium.com/master-tips-for-amazon-quicksight-spice-dataset-direct-query-mode-to-boost-your-analytics-game-d7dea5ba51de>