

SPRINT 6 - SALES UPDATES

1. Adding the fulfillment channel dropdown in our dashboard

We will need to capture the fulfillment channel data as well in the query so adding that parameter.

Before that checking distinct :

```
SELECT DISTINCT(fulfillment_channel)
FROM `bigqueryexport-183608.amazon.customer_order_metrics`
WHERE EXTRACT(YEAR FROM purchase_date) IN (2022,2023) AND ORDER_STATUS NOT IN
('Pending', 'Cancelled')
LIMIT 100;
```

JOB INFORMATION		RESULTS	JSC
Row	fulfillment_channel		
1	Amazon		
2	Merchant		

That is what Gregor also wants so no need to filter in the main query.

2. For scorecard, order item should be count(amazon_order_id)

Adding that also in the query.

Updated Query for 1 and 2:

```
SELECT SELLER_ID ,DATE_OF_PURCHASE, AMAZON_ORDER_ID, SALES_FULFILLMENT_CHANNEL,
ORDER_STATUS, SALES, TOTAL_ORDERS, UNITS_ORDERED
FROM
(SELECT MP_SUP_KEY AS SELLER_ID , purchase_day AS DATE_OF_PURCHASE, amazon_order_id AS
AMAZON_ORDER_ID, fulfillment_channel as SALES_FULFILLMENT_CHANNEL, ORDER_STATUS,
SUM(ITEM_PRICE * QUANTITY) OVER (PARTITION BY PURCHASE_DAY, MP_SUP_KEY,
amazon_order_id ) AS SALES,
COUNT(amazon_order_id) OVER (PARTITION BY MP_SUP_KEY) AS TOTAL_ORDERS,
SUM(QUANTITY) OVER (PARTITION BY PURCHASE_DAY, MP_SUP_KEY , amazon_order_id ) AS
UNITS_ORDERED,
```

For checking if the data is correct , checking for one seller.

Results per page: 50 ▼ 1 – 50 of 7402 |< <

But this is always going to be duplicated in the data so in my opinion, for score cards, blend data in the dashboard.

3. So separate query to keep a track for all scorecards:

```
SELECT SELLER_ID , AMAZON_ORDER_ID, SALES_FULFILLMENT_CHANNEL, ORDER_STATUS,
TOTAL_SALES, TOTAL_ORDERS, UNITS_ORDERED
FROM
(SELECT MP_SUP_KEY AS SELLER_ID , amazon_order_id AS AMAZON_ORDER_ID,
fulfillment_channel as SALES_FULFILLMENT_CHANNEL, ORDER_STATUS,
SUM(ITEM_PRICE * QUANTITY) OVER (PARTITION BY MP_SUP_KEY) AS TOTAL_SALES,
COUNT(amazon_order_id) OVER (PARTITION BY MP_SUP_KEY) AS TOTAL_ORDERS,
SUM(QUANTITY) OVER (PARTITION BY MP_SUP_KEY , amazon_order_id ) AS UNITS_ORDERED,
ROW_NUMBER() OVER (PARTITION BY MP_SUP_KEY) as rn
FROM `bigqueryexport-183608.amazon.customer_order_metrics`
WHERE EXTRACT(YEAR FROM purchase_date) IN (2022,2023) AND ORDER_STATUS NOT IN
('Pending', 'Cancelled'))
WHERE rn=1;
```

Manually Checking for seller - b0579da9-b708-4472-9e3b-2b2b08aee618 with google sheets:

Total sales score card - 2,685,126.53 CORRECT

Total orders- 41014 CORRECT (Duplicates there in data - check) (LATER)

Units Ordered - 49251 WRONG

Reworking on the query -

```
SELECT SELLER_ID , TOTAL_SALES, TOTAL_ORDERS, UNITS_ORDERED
FROM
(SELECT MP_SUP_KEY AS SELLER_ID , amazon_order_id AS AMAZON_ORDER_ID,
SUM(ITEM_PRICE * QUANTITY) OVER (PARTITION BY MP_SUP_KEY) AS TOTAL_SALES,
COUNT(amazon_order_id) OVER (PARTITION BY MP_SUP_KEY) AS TOTAL_ORDERS,
SUM(QUANTITY) OVER (PARTITION BY MP_SUP_KEY ) AS UNITS_ORDERED,
ROW_NUMBER() OVER (PARTITION BY MP_SUP_KEY) as rn
FROM `bigqueryexport-183608.amazon.customer_order_metrics`
WHERE EXTRACT(YEAR FROM purchase_date) IN (2022,2023) AND ORDER_STATUS NOT IN
('Pending', 'Cancelled'))
WHERE rn=1;
```

WORKEDD !!!

4. Now checking if it works on simple dashboard: (Also check another scorecards)

d3727236-53ab-45fc-81a2-bba077c33074

Gregor -

Sales Snapshot <small>taken at 3/27/2023, 2:10:30 PM PDT</small>				
Total order items	Units ordered	Ordered product sales	Avg. units/order item	Avg. sales/order item
143,622	155,101	\$5,021,868.50	1.08	\$34.97
Compare Sales			Graph view	Table view

customer_order_metrics

SELLER_ID: d3727236-53ab-45fc-81... (1) ▼

TOTAL_ORDERS
1,213,763

UNITS_ORDERED
1,312,520

TOTAL_SALES
60,080,235.78

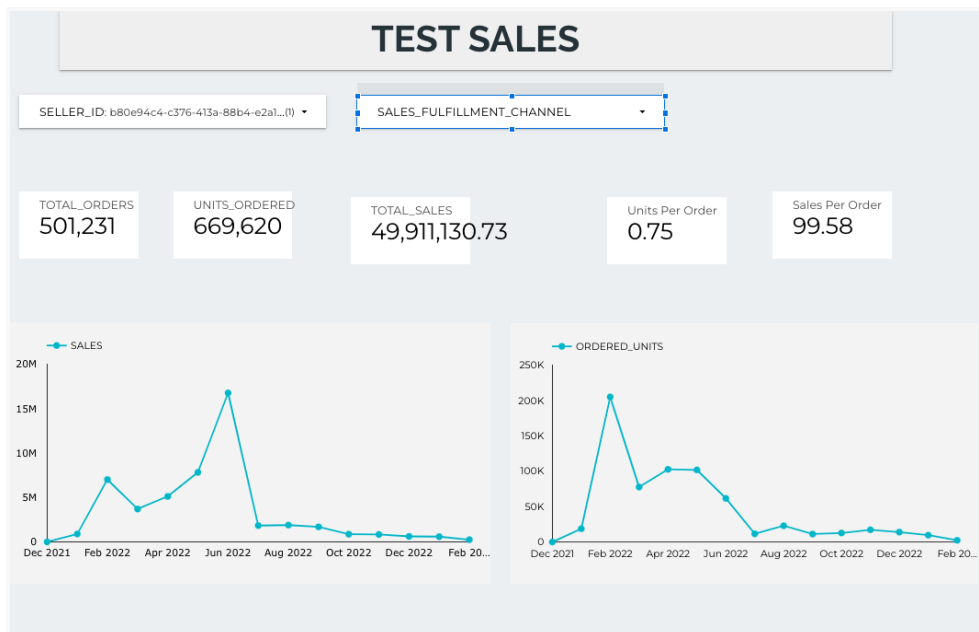
Units per order
1.08

Sales per order
49.5

Nearly similar.

5. Validating for customer 1 : b80e94c4-c376-413a-88b4-e2a1ddd980d9

New Dashboard



Too much duplicate data

Lets verify then.

2022-02-28	111-2265310-0321055
------------	---------------------

SALES- 202.28 SAME
ORDERED-UNITS- 26 SAME

Sum	202.28 +
-----	----------

So why is this inflated?

PREVIEW																	
Tr	Tr	123	123	Tr	Tr	Tr	Tr	Tr	Tr	Tr	Tr	Tr	Tr	Tr	Tr	Tr	Tr
amazon_o	order_stat	item_price	quantity	last_updat	fulfillment	ship_servi	asin	sku	sales_cha	purchase	purchase	is_fba	mp_sup_k	partition_c	path_gold	SALES	
111-2265310-03	Shipped	7.78	1	3/1/2022 14:21	Merchant	FreeEconomy	B006MWW02S	PA GLU24233	Amazon.com	3/1/2022 0:02:3	2/28/2022	FALSE	b80e94c4-c376-2022-03-30	schema-amazon		7.78	
111-2265310-03	Shipped	7.78	1	3/1/2022 14:21	Merchant	FreeEconomy	B006MWW02S	PA GLU24233	Amazon.com	3/1/2022 0:02:3	2/28/2022	FALSE	b80e94c4-c376-2022-03-02	schema-amazon		7.78	
111-2265310-03	Shipped	7.78	1	3/1/2022 14:21	Merchant	FreeEconomy	B006MWW02S	PA GLU24233	Amazon.com	3/1/2022 0:02:3	2/28/2022	FALSE	b80e94c4-c376-2022-03-09	schema-amazon		7.78	
111-2265310-03	Shipped	7.78	1	3/1/2022 14:21	Merchant	FreeEconomy	B006MWW02S	PA GLU24233	Amazon.com	3/1/2022 0:02:3	2/28/2022	FALSE	b80e94c4-c376-2022-03-04	schema-amazon		7.78	
111-2265310-03	Shipped	7.78	1	3/1/2022 14:21	Merchant	FreeEconomy	B006MWW02S	PA GLU24233	Amazon.com	3/1/2022 0:02:3	2/28/2022	FALSE	b80e94c4-c376-2022-03-07	schema-amazon		7.78	
111-2265310-03	Shipped	7.78	1	3/1/2022 14:21	Merchant	FreeEconomy	B006MWW02S	PA GLU24233	Amazon.com	3/1/2022 0:02:3	2/28/2022	FALSE	b80e94c4-c376-2022-03-21	schema-amazon		7.78	
111-2265310-03	Shipped	7.78	1	3/1/2022 14:21	Merchant	FreeEconomy	B006MWW02S	PA GLU24233	Amazon.com	3/1/2022 0:02:3	2/28/2022	FALSE	b80e94c4-c376-2022-03-05	schema-amazon		7.78	
111-2265310-03	Shipped	7.78	1	3/1/2022 14:21	Merchant	FreeEconomy	B006MWW02S	PA GLU24233	Amazon.com	3/1/2022 0:02:3	2/28/2022	FALSE	b80e94c4-c376-2022-03-03	schema-amazon		7.78	
111-2265310-03	Shipped	7.78	1	3/1/2022 14:21	Merchant	FreeEconomy	B006MWW02S	PA GLU24233	Amazon.com	3/1/2022 0:02:3	2/28/2022	FALSE	b80e94c4-c376-2022-03-08	schema-amazon		7.78	
111-2265310-03	Shipped	7.78	1	3/1/2022 14:21	Merchant	FreeEconomy	B006MWW02S	PA GLU24233	Amazon.com	3/1/2022 0:02:3	2/28/2022	FALSE	b80e94c4-c376-2022-03-06	schema-amazon		7.78	
111-2265310-03	Shipped	7.78	1	3/1/2022 14:21	Merchant	FreeEconomy	B006MWW02S	PA GLU24233	Amazon.com	3/1/2022 0:02:3	2/28/2022	FALSE	b80e94c4-c376-2022-03-23	schema-amazon		7.78	
111-2265310-03	Shipped	7.78	1	3/1/2022 14:21	Merchant	FreeEconomy	B006MWW02S	PA GLU24233	Amazon.com	3/1/2022 0:02:3	2/28/2022	FALSE	b80e94c4-c376-2022-03-26	schema-amazon		7.78	
111-2265310-03	Shipped	7.78	1	3/1/2022 14:21	Merchant	FreeEconomy	B006MWW02S	PA GLU24233	Amazon.com	3/1/2022 0:02:3	2/28/2022	FALSE	b80e94c4-c376-2022-03-25	schema-amazon		7.78	
111-2265310-03	Shipped	7.78	1	3/1/2022 14:21	Merchant	FreeEconomy	B006MWW02S	PA GLU24233	Amazon.com	3/1/2022 0:02:3	2/28/2022	FALSE	b80e94c4-c376-2022-03-16	schema-amazon		7.78	
111-2265310-03	Shipped	7.78	1	3/1/2022 14:21	Merchant	FreeEconomy	B006MWW02S	PA GLU24233	Amazon.com	3/1/2022 0:02:3	2/28/2022	FALSE	b80e94c4-c376-2022-03-18	schema-amazon		7.78	
111-2265310-03	Shipped	7.78	1	3/1/2022 14:21	Merchant	FreeEconomy	B006MWW02S	PA GLU24233	Amazon.com	3/1/2022 0:02:3	2/28/2022	FALSE	b80e94c4-c376-2022-03-12	schema-amazon		7.78	
111-2265310-03	Shipped	7.78	1	3/1/2022 14:21	Merchant	FreeEconomy	B006MWW02S	PA GLU24233	Amazon.com	3/1/2022 0:02:3	2/28/2022	FALSE	b80e94c4-c376-2022-03-13	schema-amazon		7.78	
111-2265310-03	Shipped	7.78	1	3/1/2022 14:21	Merchant	FreeEconomy	B006MWW02S	PA GLU24233	Amazon.com	3/1/2022 0:02:3	2/28/2022	FALSE	b80e94c4-c376-2022-03-15	schema-amazon		7.78	
111-2265310-03	Shipped	7.78	1	3/1/2022 14:21	Merchant	FreeEconomy	B006MWW02S	PA GLU24233	Amazon.com	3/1/2022 0:02:3	2/28/2022	FALSE	b80e94c4-c376-2022-03-11	schema-amazon		7.78	
111-2265310-03	Shipped	7.78	1	3/1/2022 14:21	Merchant	FreeEconomy	B006MWW02S	PA GLU24233	Amazon.com	3/1/2022 0:02:3	2/28/2022	FALSE	b80e94c4-c376-2022-03-17	schema-amazon		7.78	
111-2265310-03	Shipped	7.78	1	3/1/2022 14:21	Merchant	FreeEconomy	B006MWW02S	PA GLU24233	Amazon.com	3/1/2022 0:02:3	2/28/2022	FALSE	b80e94c4-c376-2022-03-10	schema-amazon		7.78	
111-2265310-03	Shipped	7.78	1	3/1/2022 14:21	Merchant	FreeEconomy	B006MWW02S	PA GLU24233	Amazon.com	3/1/2022 0:02:3	2/28/2022	FALSE	b80e94c4-c376-2022-03-24	schema-amazon		7.78	
111-2265310-03	Shipped	7.78	1	3/1/2022 14:21	Merchant	FreeEconomy	B006MWW02S	PA GLU24233	Amazon.com	3/1/2022 0:02:3	2/28/2022	FALSE	b80e94c4-c376-2022-03-20	schema-amazon		7.78	
111-2265310-03	Shipped	7.78	1	3/1/2022 14:21	Merchant	FreeEconomy	B006MWW02S	PA GLU24233	Amazon.com	3/1/2022 0:02:3	2/28/2022	FALSE	b80e94c4-c376-2022-03-22	schema-amazon		7.78	
111-2265310-03	Shipped	7.78	1	3/1/2022 14:21	Merchant	FreeEconomy	B006MWW02S	PA GLU24233	Amazon.com	3/1/2022 0:02:3	2/28/2022	FALSE	b80e94c4-c376-2022-03-28	schema-amazon		7.78	

All are duplicate rows just the partition_date is different.

So you will have to have a filter on this partition date to remove these inflated values

Understanding Partition_Date:

```
SELECT DISTINCT(partition_date)
```

```
FROM `bigqueryexport-183608.amazon.customer_order_metrics`
```

```
WHERE EXTRACT(YEAR FROM purchase_date) IN (2022, 2023) AND ORDER_STATUS NOT IN ('Pending', 'Cancelled')
```

```
ORDER BY partition_date DESC;
```

JOB INFORMATION		RESULTS
Row	partition_date	
1	2023-04-02	
2	2023-04-01	
3	2023-03-31	
4	2023-03-30	
5	2023-03-29	
6	2023-03-28	
7	2023-03-27	
8	2023-03-26	
9	2023-03-25	
10	2023-03-24	
11	2023-03-23	
12	2023-03-22	
13	2023-03-21	
14	2023-03-20	
15	2023-03-19	
16	2023-03-18	
17	2023-03-17	

In Row_number - adding the partition date as well.

```

SELECT SELLER_ID ,DATE_OF_PURCHASE, AMAZON_ORDER_ID,  SALES_FULFILLMENT_CHANNEL,
ORDER_STATUS, SALES, ORDERED_UNITS,
FROM
(SELECT MP_SUP_KEY AS SELLER_ID ,purchase_day AS DATE_OF_PURCHASE, amazon_order_id AS
AMAZON_ORDER_ID, fulfillment_channel as SALES_FULFILLMENT_CHANNEL, ORDER_STATUS,
SUM(ITEM_PRICE * QUANTITY) OVER (PARTITION BY PURCHASE_DAY, MP_SUP_KEY,
amazon_order_id ) AS SALES,
SUM(QUANTITY) OVER (PARTITION BY PURCHASE_DAY, MP_SUP_KEY , amazon_order_id ) AS
ORDERED_UNITS,
ROW_NUMBER() OVER (PARTITION BY PURCHASE_DAY, MP_SUP_KEY, amazon_order_id,
partition_date) as rn
FROM `bigqueryexport-183608.amazon.customer_order_metrics`
WHERE EXTRACT(YEAR FROM purchase_date) IN (2022,2023) AND ORDER_STATUS NOT IN
('Pending', 'Cancelled'))

```

WHERE rn=1 AND SELLER_ID= 'b80e94c4-c376-413a-88b4-e2a1ddd980d9' ;

Getting that 26 times.

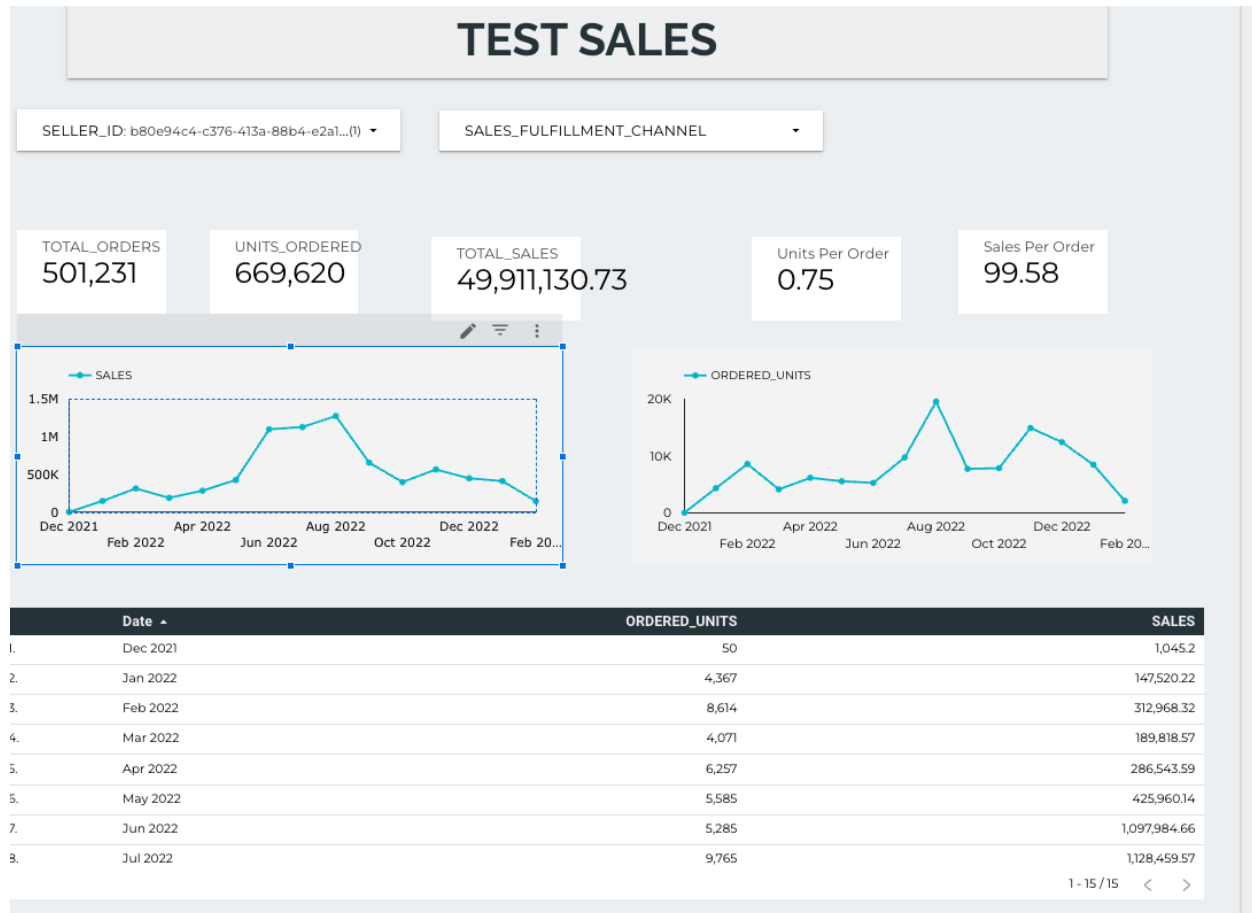
WRONG. Infact worse. Issue is I am adding in row_numeber line. That is wrong. Add instead to sum statements.

```
SELECT SELLER_ID ,DATE_OF_PURCHASE, AMAZON_ORDER_ID, SALES_FULFILLMENT_CHANNEL,
ORDER_STATUS, SALES, ORDERED_UNITS,
FROM
(SELECT MP_SUP_KEY AS SELLER_ID ,purchase_day AS DATE_OF_PURCHASE, amazon_order_id AS
AMAZON_ORDER_ID, fulfillment_channel as SALES_FULFILLMENT_CHANNEL, ORDER_STATUS,
SUM(ITEM_PRICE * QUANTITY) OVER (PARTITION BY PURCHASE_DAY, MP_SUP_KEY,
amazon_order_id, partition_date ) AS SALES,
SUM(QUANTITY) OVER (PARTITION BY PURCHASE_DAY, MP_SUP_KEY , amazon_order_id,
partition_date ) AS ORDERED_UNITS,
ROW_NUMBER() OVER (PARTITION BY PURCHASE_DAY, MP_SUP_KEY, amazon_order_id) as rn
FROM `bigqueryexport-183608.amazon.customer_order_metrics`
WHERE EXTRACT(YEAR FROM purchase_date) IN (2022,2023) AND ORDER_STATUS NOT IN
('Pending', 'Cancelled'))
WHERE rn=1 AND SELLER_ID= 'b80e94c4-c376-413a-88b4-e2a1ddd980d9' AND AMAZON_ORDER_ID=
'111-2265310-0321055' ;
```

Query results SAVE RESU

JOB INFORMATION		RESULTS	JSON	EXECUTION DETAILS	EXECUTION GRAPH	PREVIEW		
Row	SELLER_ID	DATE_OF_PURCH	AMAZON_ORDER_ID	SALES_FULFILLMENT_CHANNEL	ORDER_STATUS	SALES	ORDERED_UNITS	
1	b80e94c4-c376-413a-88b4-e2a...	2022-02-28	111-2265310-0321055	Merchant	Shipped	7.78	1	

Looks correct. Check once.



Decreased and looks much better now. Still not there.

Look again why. So CHECK FOR ONE MONTH AGAIN.

```

SELECT MP_SUP_KEY AS SELLER_ID ,purchase_day AS DATE_OF_PURCHASE, amazon_order_id AS
AMAZON_ORDER_ID, fulfillment_channel as SALES_FULFILLMENT_CHANNEL, ORDER_STATUS,
partition_date,
SUM(ITEM_PRICE * QUANTITY) OVER (PARTITION BY PURCHASE_DAY, MP_SUP_KEY,
amazon_order_id, partition_date ) AS SALES,
SUM(QUANTITY) OVER (PARTITION BY PURCHASE_DAY, MP_SUP_KEY , amazon_order_id,
partition_date ) AS ORDERED_UNITS,
ROW_NUMBER() OVER (PARTITION BY PURCHASE_DAY, MP_SUP_KEY, amazon_order_id) as rn
FROM `bigqueryexport-183608.amazon.customer_order_metrics`
WHERE EXTRACT(YEAR FROM purchase_date) IN (2022) AND EXTRACT(MONTH FROM purchase_date)
IN (2) AND ORDER_STATUS NOT IN ('Pending', 'Cancelled') and
MP_SUP_KEY='b80e94c4-c376-413a-88b4-e2a1ddd980d9' ;

```


2022-02-28	114-5554790-0058621
------------	---------------------

Correct.

item_price //	quantity //
68.52	2

26 records

//	SALES //	ORDERED_UNITS //	partition_date //
	137.04	2	2022-03-23

1 record only

Check total for a month - query matches with dashboard. Order ids also in google sheets- 1.

In ACTUAL data for the month - 6645 unique order ids.
My query - 6645 CORRECT

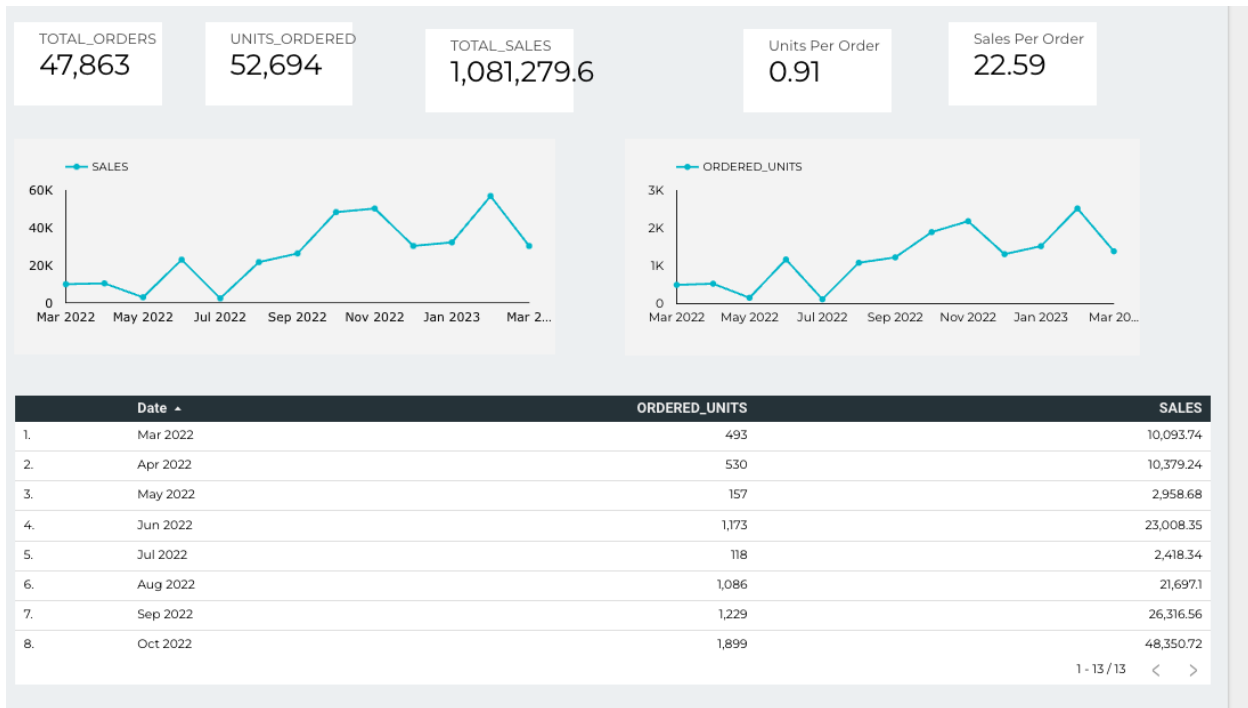
Rn one exploring with sheets and filtering rn=1
In actual data - sales - sum - 310,966.89, orders- 8525
My query - sum - 312,968.32 , orders- 8614

ORDER BOTH BY ORDERID TO SEE FIRST 10 RECORDS IN BOTH THE SHEETS

SALES	C
14.95	
42.66	
9.05	
49.84	
30.52	
18.33	
18.11	
26.51	
12.35	
79.02	
7.69	
13.95	
7.45	
18.11	
7	

Same in Google sheets too.

Validating for customer 2 : 1ee02181-2249-4cc9-92c4-c988c5f8b029



Trend same followed. 2-3K extra in sales for 4-5 months that I checked.
Scorecard last three values similar. First two no.

MAde the first two score_Cards with OG data. - closer to data and similar to actual unique values in data

AVERAGE V/S AVERAGE performance every month

```

SELECT SELLER_ID , CONCAT(CAST(MONTH AS STRING), '-' , CAST(YEAR AS STRING)) AS
MONTH_YEAR,
DATE_OF_PURCHASE, amazon_order_id, ORDER_STATUS, AVG_SALES_SELLER, TOTAL_SALES,
COUNT_SELLERS, AVG_TOTAL_SALES
FROM
(SELECT EXTRACT(MONTH FROM purchase_day) AS MONTH,
EXTRACT(YEAR FROM purchase_day) AS YEAR,
MP_SUP_KEY AS SELLER_ID ,purchase_day AS DATE_OF_PURCHASE, amazon_order_id,
ORDER_STATUS, partition_date,
AVG(ITEM_PRICE * QUANTITY) OVER (PARTITION BY EXTRACT(MONTH FROM purchase_day)
,EXTRACT(YEAR FROM purchase_day), mp_sup_key) AS AVG_SALES_SELLER,
SUM(ITEM_PRICE * QUANTITY) OVER (PARTITION BY EXTRACT(MONTH FROM purchase_day) ,
EXTRACT(YEAR FROM purchase_day)) AS TOTAL_SALES,

```

```

COUNT(MP_SUP_KEY) OVER (PARTITION BY EXTRACT(MONTH FROM purchase_day) , EXTRACT(YEAR
FROM purchase_day)) AS COUNT_SELLERS,
CAST((SUM(ITEM_PRICE * QUANTITY) OVER (PARTITION BY EXTRACT(MONTH FROM purchase_day) ,
EXTRACT(YEAR FROM purchase_day))/ COUNT(MP_SUP_KEY) OVER (PARTITION BY EXTRACT(MONTH
FROM purchase_day) , EXTRACT(YEAR FROM purchase_day)) ) AS FLOAT64) AS
AVG_TOTAL_SALES,
ROW_NUMBER() OVER (PARTITION BY EXTRACT(MONTH FROM purchase_day) , EXTRACT(YEAR FROM
purchase_day)) as row_day
FROM `bigqueryexport-183608.amazon.customer_order_metrics`
WHERE EXTRACT(YEAR FROM purchase_date) IN (2022,2023) AND ORDER_STATUS NOT IN
('Pending', 'Cancelled'))
WHERE row_day=1;

```

Tried working on comparing the average with the average but did not work.

Selecting only few sellers because row_number partition issue..

```

SELECT SELLER_ID , CONCAT(CAST(MONTH AS STRING), '-' , CAST(YEAR AS STRING)) AS
MONTH_YEAR,
DATE_OF_PURCHASE, amazon_order_id, ORDER_STATUS, AVG_SALES_SELLER, TOTAL_SALES,
COUNT_SELLERS, AVG_TOTAL_SALES
FROM
(SELECT EXTRACT(MONTH FROM purchase_day) AS MONTH,
EXTRACT(YEAR FROM purchase_day) AS YEAR,
MP_SUP_KEY AS SELLER_ID ,purchase_day AS DATE_OF_PURCHASE, amazon_order_id,
ORDER_STATUS, partition_date,
AVG(ITEM_PRICE * QUANTITY) OVER (PARTITION BY EXTRACT(MONTH FROM purchase_day)
,EXTRACT(YEAR FROM purchase_day), mp_sup_key) AS AVG_SALES_SELLER,
SUM(ITEM_PRICE * QUANTITY) OVER (PARTITION BY EXTRACT(MONTH FROM purchase_day) ,
EXTRACT(YEAR FROM purchase_day)) AS TOTAL_SALES,
COUNT(MP_SUP_KEY) OVER (PARTITION BY EXTRACT(MONTH FROM purchase_day) , EXTRACT(YEAR
FROM purchase_day)) AS COUNT_SELLERS,
CAST((SUM(ITEM_PRICE * QUANTITY) OVER (PARTITION BY EXTRACT(MONTH FROM purchase_day) ,
EXTRACT(YEAR FROM purchase_day))/ COUNT(MP_SUP_KEY) OVER (PARTITION BY EXTRACT(MONTH
FROM purchase_day) , EXTRACT(YEAR FROM purchase_day)) ) AS FLOAT64) AS
AVG_TOTAL_SALES,

```

```

ROW_NUMBER() OVER (PARTITION BY EXTRACT(MONTH FROM purchase_day) , EXTRACT(YEAR FROM
purchase_day), mp_sup_key) as row_day
FROM `bigqueryexport-183608.amazon.customer_order_metrics`
WHERE EXTRACT(YEAR FROM purchase_date) IN (2022,2023) AND ORDER_STATUS NOT IN
('Pending', 'Cancelled'))
WHERE row_day=1;

```

COMPETITIVE PRICING

new_buy_box_landed_price_amount:

The "new_buy_box_landed_price_amount" on Amazon refers to the current lowest price (including shipping costs) that a customer can purchase a new item for from a seller who has won the Buy Box. The Buy Box is the section on an Amazon product detail page where customers can click "Add to Cart" or "Buy Now" to make a purchase.

The "new_buy_box_landed_price_amount" specifically refers to the landed price of the item, which includes the price of the product plus any shipping and handling costs. This is the total amount that the customer will pay to have the item delivered to them.

The Buy Box is highly sought after by sellers on Amazon, as it greatly increases the visibility and likelihood of making a sale. To win the Buy Box, a seller must meet certain criteria, such as having competitive pricing, good seller ratings, and fast and reliable shipping times.

New_buy_box_listing_price_amount:

The "new_buy_box_listing_price_amount" on Amazon refers to the current price that a seller has listed a new item for on Amazon. This is the price that the seller has set for the product before any shipping or handling costs are added.

When multiple sellers are offering the same product on Amazon, the seller with the lowest "new_buy_box_listing_price_amount" will typically win the Buy Box, which is the section on an Amazon product detail page where customers can click "Add to Cart" or "Buy Now" to make a purchase.

It's worth noting that while the "new_buy_box_listing_price_amount" is an important factor in winning the Buy Box, it's not the only one. Other factors such as the seller's performance metrics (such as shipping times, customer service, and return rates) and the item's availability and condition are also taken into account.

New_buy_box_shipping_price_amount:

The "new_buy_box_shipping_price_amount" on Amazon refers to the current shipping price that a customer would pay to have a new item shipped to them from the seller who has won the Buy

Box. This is in addition to the "new_buy_box_listing_price_amount", which is the price that the seller has listed the item for.

The "new_buy_box_shipping_price_amount" can vary depending on a number of factors, such as the shipping destination, the size and weight of the item, and the shipping method selected by the customer. In some cases, the seller may offer free shipping, in which case the "new_buy_box_shipping_price_amount" would be zero.

When multiple sellers are offering the same product on Amazon, the seller with the lowest combined "new_buy_box_listing_price_amount" and "new_buy_box_shipping_price_amount" will typically win the Buy Box, which is the section on an Amazon product detail page where customers can click "Add to Cart" or "Buy Now" to make a purchase.

ALL OF IT SIMILARLY FOR THE USED ITEMS.

listing_count_offers:

"listing_count_offers" on Amazon refers to the total number of offers (i.e. listings) for a particular product on the Amazon marketplace. An offer is created by a seller and represents a unique listing of a product, which may include different prices, conditions, and shipping options.

The "listing_count_offers" metric can provide valuable information for both buyers and sellers. For buyers, it can help them to quickly assess the level of competition and the range of prices and conditions available for a particular product. For sellers, it can help them to gauge the level of competition and adjust their pricing and other offers details to stand out from the crowd.

In general, products with a higher "listing_count_offers" tend to be more competitive, while products with a lower "listing_count_offers" may have less competition and may be easier for sellers to win the Buy Box. However, the level of competition can vary depending on a number of factors, such as the popularity of the product, the time of year, and the specific conditions and prices offered by each seller.

Distinct Partition Date:
earliest latest is in 2022.

Row	partition_date
1	2022-08-28 00:00:00 UTC
2	2022-08-21 00:00:00 UTC
3	2022-08-14 00:00:00 UTC
4	2022-08-07 00:00:00 UTC
5	2022-07-31 00:00:00 UTC
6	2022-07-24 00:00:00 UTC
7	2022-07-17 00:00:00 UTC
8	2022-07-10 00:00:00 UTC
9	2022-07-03 00:00:00 UTC
10	2022-06-26 00:00:00 UTC
11	2022-06-19 00:00:00 UTC
12	2022-06-12 00:00:00 UTC

1. For an asin, checking the listing_count_offers, but partitioning by partition_Date:

```
SELECT STATUS, ASIN, new_buy_box_landed_price_amount,
new_buy_box_landed_price_currency_code,listing_count_offers,
ROW_NUMBER() OVER (PARTITION BY asin, partition_date) as rn
FROM `bigqueryexport-183608.amazon.competitive_pricing`
WHERE EXTRACT(YEAR FROM partition_date) IN (2022,2023);
```

Exploring data with sheets:

Multiple values for one asin.

Filtering on ASIN = B07YNR7S5S

T _T	T _T	T _T	T _T	T _T
STATUS	ASIN	new_buy_	new_buy_	listing_count_offers
Success	B07YNR7S5S	37.90	USD	2
Success	B07YNR7S5S	34.90	USD	2
Success	B07YNR7S5S	37.90	USD	2
Success	B07YNR7S5S	39.70	USD	2
Success	B07YNR7S5S	35.90	USD	2
Success	B07YNR7S5S	34.90	USD	2
Success	B07YNR7S5S	39.70	USD	2
Success	B07YNR7S5S	39.70	USD	2
Success	B07YNR7S5S	37.90	USD	2
Success	B07YNR7S5S	34.90	USD	2
Success	B07YNR7S5S	34.90	USD	2
Success	B07YNR7S5S	39.70	USD	2
Success	B07YNR7S5S	35.90	USD	2
Success	B07YNR7S5S	39.70	USD	2
Success	B07YNR7S5S	34.90	USD	2
Success	B07YNR7S5S	37.90	USD	2
Success	B07YNR7S5S	37.90	USD	2
Success	B07YNR7S5S	37.90	USD	2
Success	B07YNR7S5S	34.90	USD	2
Success	B07YNR7S5S	34.90	USD	2
Success	B07YNR7S5S	39.70	USD	2

Yes so values are different for an asin for a different partition date. Maybe more demand , increases.

General:

Listing_count_offers range from 1-1709.

To see the natural grouping pattern - converting into number and plotting histogram

```
SELECT ASIN, STATUS, new_buy_box_landed_price_amount,
new_buy_box_landed_price_currency_code,listing_count_offers, partition_date
FROM
(SELECT ASIN, STATUS, new_buy_box_landed_price_amount,
new_buy_box_landed_price_currency_code,SAFE_CAST("listing_count_offers" AS INT64) as
listing_count_offers, DATE(partition_date) as partition_date,
```





```

ROW_NUMBER() OVER (PARTITION BY asin, partition_date) as rn
FROM `bigqueryexport-183608.amazon.competitive_pricing`
WHERE EXTRACT(YEAR FROM partition_date) IN (2022,2023))
WHERE rn=1
ORDER BY listing_count_offers DESC;

```

Average	12.04	+
Median	5	
Minimum value	1	+
Maximum value	1,709	+

 Most  Least	
VALUE	FREQUENCY
1	398,020
2	172,070
3	105,729
4	88,228
5	78,014

Insert pivot table

<div> <div>↗ Most</div> <div>↘ Least</div> </div>	
VALUE	FREQUENCY
546	1
789	1
832	1
827	1
977	1
<div>Insert pivot table</div>	

4 categories-

No competition 1

Medium competition 2-4

High competition- 5-20

Outliers > 20

```

SELECT ASIN, STATUS, new_buy_box_landed_price_amount,
new_buy_box_landed_price_currency_code, partition_date, listing_count_offers_number,
category ,
FROM
(SELECT ASIN, STATUS, new_buy_box_landed_price_amount,
new_buy_box_landed_price_currency_code,
SAFE_CAST(listing_count_offers AS INT64) as listing_count_offers_number,
DATE(partition_date) as partition_date,
ROW_NUMBER() OVER (PARTITION BY asin, partition_date) as rn,
CASE
WHEN SAFE_CAST(listing_count_offers AS INT64) = 1 THEN 'No Competition'
WHEN SAFE_CAST(listing_count_offers AS INT64) IN (2,3, 4) THEN 'Medium Competition'
WHEN SAFE_CAST(listing_count_offers AS INT64) >=5 and SAFE_CAST(listing_count_offers
AS INT64) <=20 THEN 'High Competition'
WHEN SAFE_CAST(listing_count_offers AS INT64) >21 THEN 'High Outliers'
END AS category

```

```
FROM `bigqueryexport-183608.amazon.competitive_pricing`  
WHERE EXTRACT(YEAR FROM partition_date) IN (2022,2023))  
WHERE rn=1;
```

Took many tries to form this query !

Issue was coming while plotting the price on dashboard wrt time..

So converted the datatype of landed price currency...

```
SELECT ASIN, STATUS, new_buy_box_landed_price_amount,  
new_buy_box_landed_price_currency_code, partition_date, listing_count_offers_number,  
category ,  
FROM  
(SELECT ASIN, STATUS, SAFE_CAST(new_buy_box_landed_price_amount AS FLOAT64) AS  
new_buy_box_landed_price_amount, new_buy_box_landed_price_currency_code,  
SAFE_CAST(listing_count_offers AS INT64) as listing_count_offers_number,  
DATE(partition_date) as partition_date,  
ROW_NUMBER() OVER (PARTITION BY asin, partition_date) as rn,  
CASE  
WHEN SAFE_CAST(listing_count_offers AS INT64) = 1 THEN 'No Competition'  
WHEN SAFE_CAST(listing_count_offers AS INT64) IN (2,3, 4) THEN 'Medium Competition'  
WHEN SAFE_CAST(listing_count_offers AS INT64) >=5 and SAFE_CAST(listing_count_offers  
AS INT64) <=20 THEN 'High Competition'  
WHEN SAFE_CAST(listing_count_offers AS INT64) >21 THEN 'High Outliers'  
END AS category  
FROM `bigqueryexport-183608.amazon.competitive_pricing`  
WHERE EXTRACT(YEAR FROM partition_date) IN (2022,2023))  
WHERE rn=1;
```

COMPETITIVE PRICING

ASIN

ASIN

481,147

listing_count_offers_number

19,107,122

