BASKET SIZE

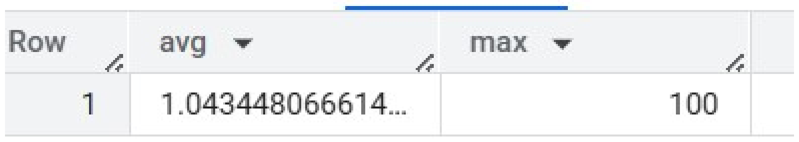
```
Select--highest basket size
distinct transaction_id,
avg (item_price) avg,
max(item_quantity) max
From prism-insights.warehouse_PT.transactionsanditems
Group By transaction_id
Order by transaction_id;

SELECT --total_spend, purchase_count, AOV, lifespan
  user_crm_id,
  SUM(transaction_total) AS total_spent,
  COUNT(DISTINCT transaction_id) AS purchase_count,
  AVG(transaction_total) AS avg_order_value,
  DATE_DIFF(MAX(date), MIN(date), DAY) AS customer_lifespan
FROM `prism-insights.warehouse_PT.transactions`
WHERE user_crm_id IS NOT NULL
GROUP BY user_crm_id
ORDER BY purchase_count;

Select -- basket size, number of purchases and AOV per member
avg (item_quantity) avg_basket_size,
sum (item_quantity) total_items_purchased,
COUNT(DISTINCT t.transaction_id) AS number_of_purchases,
AVG(transaction_total) AS avg_order_value,
t.user_crm_id
FROM `prism-insights.warehouse_PT.transactionsanditems` ti
Left JOIN `prism-insights.warehouse_PT.transactions` t
ON ti.transaction_id = t.transaction_id
Where user_crm_id is not null
group by user_crm_id


SELECT  -- items per order bins
    t.user_crm_id,
    AVG(ti.item_quantity) AS avg_basket_size,
    SUM(ti.item_quantity) AS total_items_purchased,
    COUNT(DISTINCT t.transaction_id) AS number_of_purchases,
    AVG(t.transaction_total) AS avg_order_value,
    CASE
        WHEN AVG(ti.item_quantity) >= 10 THEN '10+ items'
        WHEN AVG(ti.item_quantity) >= 5 THEN '5+ items'
        WHEN AVG(ti.item_quantity) >= 2 THEN '2+ items'
        ELSE 'Single item'
    END AS items_per_order
FROM `prism-insights.warehouse_PT.transactionsanditems` ti
LEFT JOIN `prism-insights.warehouse_PT.transactions` t
ON ti.transaction_id = t.transaction_id
WHERE t.user_crm_id IS NOT NULL
GROUP BY t.user_crm_id
ORDER BY t.user_crm_id;


-- basket size and avg basket size
Select
sum (item_quantity) basket_size,
avg(transaction_id) avg_basket_size
transaction_id
From
```



Row	avg	max
1	1.043448066614...	100

item_quantity - highest - 100(user_crm_id: 6171593)| transaction_id - 1009104177
lowest - 1
Suggested bins: 1 item, more than 2, more than 5, 10+

Possibility: AOV based on item quantity bins (as stated above)

(to get AOV will need to join with **transactions table**)

Any correlation between multi item purchases and larger AOV?

Weekend vs Weekday:

```
SELECT transaction_id, date,
       CASE WHEN EXTRACT(DAYOFWEEK FROM date) BETWEEN 2 AND 6 THEN 'Weekday'
            ELSE 'Weekend'
       END AS order_type
FROM prism-insights.warehouse_PT.transactions
Order by order_type;

SELECT count(transaction_id) total,
date,
       CASE WHEN EXTRACT(DAYOFWEEK FROM date) BETWEEN 2 AND 6 THEN 'Weekday'
            ELSE 'Weekend'
       END AS order_type
FROM prism-insights.warehouse_PT.transactions
Group by date, order_type
Order by order_type;
```

Hopefully can use this to explore any trends between weekday and weekend shopping, including seasonal trends (per month/quarter, etc)

Purchase vs Returns Ratio

(*(Product Returns + Transactions)*)

```
Select user_crm_id,return_quantity - this query help figure out which customer id
has more than 50 returns
From prism-insights.warehouse_PT.transactions t
Left join prism-insights.warehouse_PT.product_returns pr
ON t.transaction_id = pr.transaction_id
Where return_quantity > 50;
```

```
Select max (return_quantity) highest, min (return_quantity) lowest - max and min
return_quantity
From prism-insights.warehouse_PT.transactions t
Left join prism-insights.warehouse_PT.product_returns pr
ON t.transaction_id = pr.transaction_id
```

Important: item_quantity = total basket size and return_quantity = total quantity of return

Things to explore:
- partial returns vs full return and is it refund or exchange
- Add item category to the mix
- Create a query to calculate the actual ratio between total purchases and total returns per customer
- Returns percentage to evaluate any high risk customers as well as detect and product quality/ sizing issues etc

Returns:

```
SELECT  --return ratio over 30%
    t.user_crm_id,
    COUNT(DISTINCT CASE WHEN r.return_status IS NOT NULL THEN t.transaction_id END)
AS returned_orders,
    COUNT(DISTINCT t.transaction_id) AS total_orders,
    ROUND(
        COUNT(DISTINCT CASE WHEN r.return_status IS NOT NULL THEN t.transaction_id
END) * 100.0 /
        COUNT(DISTINCT t.transaction_id), 2
    ) AS return_rate_percentage
FROM `prism-insights.warehouse_PT.transactions` t
LEFT JOIN `prism-insights.warehouse_PT.product_returns` r
ON t.transaction_id = r.transaction_id
WHERE t.user_crm_id IS NOT NULL
GROUP BY t.user_crm_id
HAVING return_rate_percentage > 30
ORDER BY return_rate_percentage DESC;

SELECT  -- returned order in less than 7 days
    t.user_crm_id,
    count (distinct t.transaction_id)total_purchases,
    COUNT(DISTINCT r.return_status) return_count,
    AVG(DATE_DIFF(r.return_date, t.date, DAY)) avg_days_to_return
FROM `prism-insights.warehouse_PT.transactions` t
LEFT JOIN `prism-insights.warehouse_PT.product_returns` r
ON t.transaction_id = r.transaction_id
WHERE t.user_crm_id IS NOT NULL
GROUP BY t.user_crm_id
HAVING avg_days_to_return < 7
ORDER BY return_count DESC;

SELECT  -- return ratio over 30% and return in less than 7 days from purchase
    t.user_crm_id,
    COUNT(DISTINCT t.transaction_id) AS total_orders,
    COUNT(DISTINCT CASE WHEN r.return_status IS NOT NULL THEN t.transaction_id END)
AS returned_orders,
    ROUND(
        COUNT(DISTINCT CASE WHEN r.return_status IS NOT NULL THEN t.transaction_id
END) * 100.0 /
        COUNT(DISTINCT t.transaction_id), 2
    ) AS return_rate_percentage,
    COUNT(DISTINCT r.return_status) AS return_count,
```

```sql
    AVG(DATE_DIFF(r.return_date, t.date, DAY)) AS avg_days_to_return
FROM `prism-insights.warehouse_PT.transactions` t
LEFT JOIN `prism-insights.warehouse_PT.product_returns` r
ON t.transaction_id = r.transaction_id
WHERE t.user_crm_id IS NOT NULL
GROUP BY t.user_crm_id
HAVING return_rate_percentage > 30
AND avg_days_to_return < 7
ORDER BY return_count DESC;

SELECT  -- 30, 60, 60% return ratio and days return
    t.user_crm_id,
    COUNT(DISTINCT t.transaction_id) AS total_orders,
    COUNT(DISTINCT CASE WHEN r.return_status IS NOT NULL THEN t.transaction_id END)
AS returned_orders,
    ROUND(
        COUNT(DISTINCT CASE WHEN r.return_status IS NOT NULL THEN t.transaction_id
END) * 100.0 /
        COUNT(DISTINCT t.transaction_id), 2
    ) AS return_rate_percentage,
    COUNT(DISTINCT r.return_status) AS return_count,
    AVG(DATE_DIFF(r.return_date, t.date, DAY)) AS avg_days_to_return,
    CASE
        WHEN ROUND(
            COUNT(DISTINCT CASE WHEN r.return_status IS NOT NULL THEN
t.transaction_id END) * 100.0 /
            COUNT(DISTINCT t.transaction_id), 2
        ) >= 90 THEN '90%+ Returns'
        WHEN ROUND(
            COUNT(DISTINCT CASE WHEN r.return_status IS NOT NULL THEN
t.transaction_id END) * 100.0 /
            COUNT(DISTINCT t.transaction_id), 2
        ) >= 60 THEN '60%+ Returns'
        WHEN ROUND(
            COUNT(DISTINCT CASE WHEN r.return_status IS NOT NULL THEN
t.transaction_id END) * 100.0 /
            COUNT(DISTINCT t.transaction_id), 2
        ) >= 30 THEN '30%+ Returns'
        ELSE '<30% Returns'
    END AS return_risk_bin
FROM `prism-insights.warehouse_PT.transactions` t
LEFT JOIN `prism-insights.warehouse_PT.product_returns` r
ON t.transaction_id = r.transaction_id
WHERE t.user_crm_id IS NOT NULL
```

```sql
GROUP BY t.user_crm_id
HAVING return_rate_percentage >= 30
AND avg_days_to_return < 7
ORDER BY return_count DESC;
```

```sql
SELECT  -- aov vs basket size
    items_per_order,
    COUNT(DISTINCT transaction_id) AS total_transactions,
    SUM(transaction_total) AS total_revenue,
    AVG(transaction_total) AS avg_order_value
FROM (
    SELECT
        t.transaction_id,
        t.transaction_total,
        CASE
            WHEN SUM(ti.item_quantity) >= 10 THEN '10+ items'
            WHEN SUM(ti.item_quantity) >= 5 THEN '5+ items'
            WHEN SUM(ti.item_quantity) >= 2 THEN '2+ items'
            ELSE 'Single item'
        END AS items_per_order
    FROM `prism-insights.warehouse_PT.transactionsanditems` ti
    LEFT JOIN `prism-insights.warehouse_PT.transactions` t
    ON ti.transaction_id = t.transaction_id
    WHERE t.user_crm_id IS NOT NULL
    GROUP BY t.transaction_id, t.transaction_total
) AS categorized_orders
GROUP BY items_per_order
ORDER BY avg_order_value DESC;
```

```sql
SELECT  -- items per order bins
    t.user_crm_id,
    AVG(ti.item_quantity) AS avg_basket_size,
    SUM(ti.item_quantity) AS total_items_purchased,
    COUNT(DISTINCT t.transaction_id) AS number_of_purchases,
    AVG(t.transaction_total) AS avg_order_value,
    CASE
        WHEN SUM(ti.item_quantity) >= 2 THEN '2+ items'
        ELSE 'Single item'
    END AS items_per_order
FROM `prism-insights.warehouse_PT.transactionsanditems` ti
LEFT JOIN `prism-insights.warehouse_PT.transactions` t
```

```sql
ON ti.transaction_id = t.transaction_id
WHERE t.user_crm_id IS NOT NULL
GROUP BY t.user_crm_id
ORDER BY t.user_crm_id;
```