Urban Loom Performance Report

January 2024 - June 2024

This report analyses Urban Loom's business performance over the first six months of 2024, providing strategic insights for physical store development and overall business expansion decisions.

The main focuses from different teams have been collected, and this first report concentrates on only three of them.

1. For Jared (Product Lead): Product line optimisation Strategy.
2. For Ameena (Operations and Logistics Manager): Inventory planning for July-December 2024.
3. For Jo (Founder): Optimal physical store location strategy.

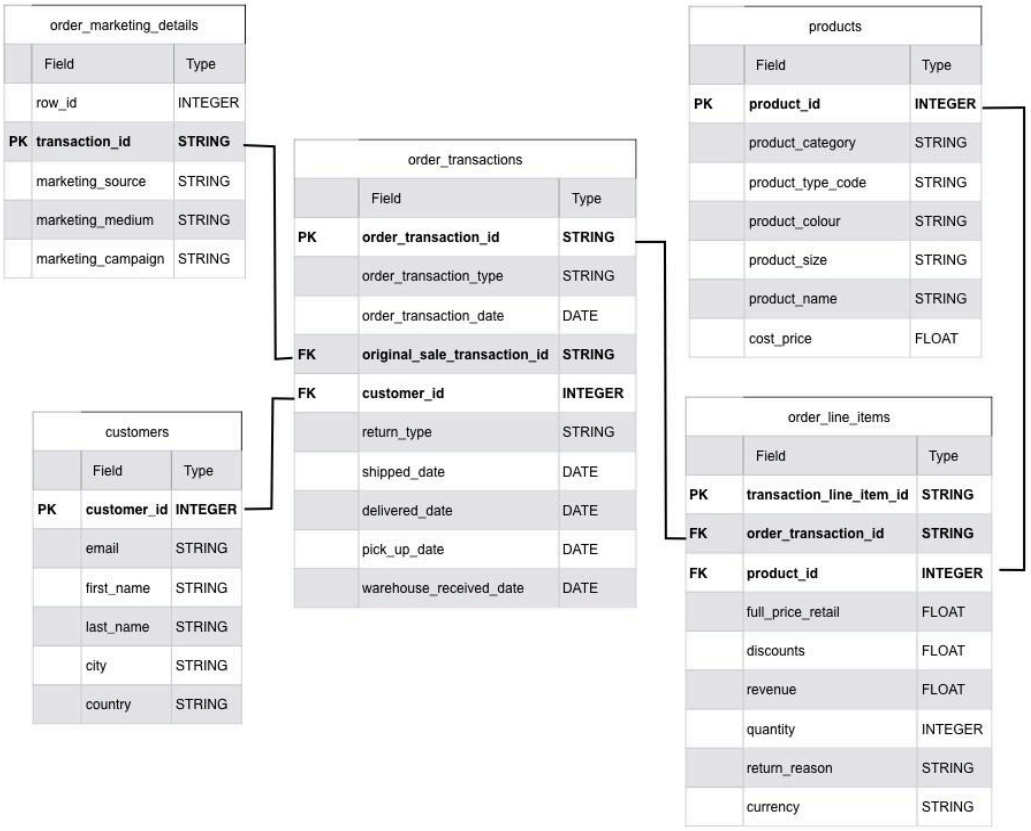

Figure 1. Logical data model illustrating the urbanloom_analytics database structure and its five interconnected tables. The primary keys were determined by the count of unique values. Data Quality Note: The products and order_transactions tables contain duplicate primary key values that require handling during analysis to ensure accurate calculations.

INSIGHT 1: Focus resources on shirts as the star category and consider discontinuing jeans from new store inventory.

1. Shirts dominate with £325,936.42 revenue (62.87% of total) and a strong 64.42% profit margin.
2. Sportswear and Outerwear round out the top three revenue generators.
3. Jeans underperformed with only £2,255.18 in revenue and the lowest profit margin at 39.95%.

Row	product_category	total_revenue	total_cost	profit_margin_per...
1	*null*	518424.75	195438.56	62.3
2	Shirts	325936.42	115974.82	64.42
3	Sportswear	91761.7	32777.7	64.28
4	Outerwear	37169.62	18520.99	50.17
5	Tops	34376.64	17606.32	48.78
6	Loungewear	18914.39	5904.86	68.78
7	Bottoms	8010.8	3299.74	58.81
8	Jeans	2255.18	1354.13	39.95

Table 1. Product performance by category, showing total revenue, total cost and profit margin, ordered from highest to lowest revenue.

```
/*
  INSIGHT 1: Product performance analysis by category
  STEP 1: Handled duplicate product_id records (from products table) and duplicate
order_transaction_id records (from order_transactions table) by keeping only the first occurrence per
transaction_line_item_id(full detail see DATA CLEANING NOTES Q1/A1).
  STEP 2: Selected records only from Jan 2024 to June 2024.
  STEP 3: Excluded gift products to avoid confusion, as they have no associated category or costs.
*/

WITH order_product_row_number AS (
        SELECT
                ROW_NUMBER() OVER (PARTITION BY oli.transaction_line_item_id ORDER BY
                oli.transaction_line_item_id) AS row_number, -- STEP 1(1)
                 *
        FROM `urbanloom_analytics.order_line_items` oli
        LEFT JOIN `urbanloom_analytics.products` p
                ON oli.product_id = p.product_id
        LEFT JOIN `urbanloom_analytics.order_transactions` ot
                ON oli.order_transaction_id = ot.order_transaction_id
        WHERE EXTRACT(MONTH FROM DATE(ot.order_transaction_date)) < 7 -- STEP 2
),
clean_order_product AS (
        SELECT *
        FROM order_product_row_number
        WHERE row_number = 1 -- STEP 1(2)
```

```
)

SELECT
        product_category,
        ROUND(SUM(revenue),2) AS total_revenue,
        ROUND(SUM(cost_price * quantity),2) AS total_cost,
        ROUND(((SUM(revenue) - SUM(cost_price * quantity))/SUM(revenue))*100, 2) AS
        profit_margin_percentage
FROM clean_order_product
WHERE product_category IS NOT NULL --STEP 3
GROUP BY ROLLUP(product_category)
ORDER BY total_revenue DESC, product_category
```

NOTE: This query includes critical data cleaning procedures, with detailed documentation of data quality issues identified and remediation methods applied.

```
/*
DATA CLEANING NOTES: Duplicate product_id and order_transaction_id entries affect the accuracy of calculations involving costs!

Q1: Why did the record count increase from 11,038 to 11,323 after JOIN?
A1: Duplicate product_id in the products table (same product, different colors) and duplicate order_transaction_id records in the order_transactions table. After inspecting the duplicate records, I believe that keeping the first record is sufficient.
SOLUTION: Remove duplicates using transaction_line_item_id (unique identifier).
REASONING STEPS: see below - Q1-A1-KEY STEP.

Q2: What are the records with NULL category?
A2: Possibly gift/promotional products in orders.
SOLUTION: Exclude from analysis (no product_category or cost_price data anyway).

Q3: SUM(revenue) vs SUM(ABS(revenue) * quantity) - which is correct?
A3: SUM(revenue) is more accurate (revenue already calculated correctly in the source)
SOLUTION: Use SUM(revenue) directly.

Q4: What are the currencies used?
A4: GBP is the only currency.

FINAL RESULT: Cleaned 10,896 records with accurate revenue/cost calculations.
*/

/*
Q1-A1-KEY STEP: Handling JOIN-induced duplicates for accurate financial calculations.

PROBLEM CONTEXT:
   1. There are duplicates on the PK in two tables:
```

```
     - products table: 839 unique product_id in total 843 records (same product_id
with different colors/variants).
     - order_transactions table: 6,281 unique order_transaction_id in total 6,351
records.
   2. When joining (LEFT JOIN) order_line_items with these two tables, the record
count increased from 11,038 to 11,323 records.This creates duplicate
transaction_line_item_id records that will cause incorrect calculations of:
     - Total revenue (sum of revenue field)
     - Total quantities (affecting cost calculations: quantity * cost_price)

EXPLORATION: Examine the duplicates in the products and order_transactions tables
separately, verify that duplicate records contain identical values for fields that
impact financial calculations.
*/

--VALIDATION 1: Do duplicate records have the identical cost_price value?
--Find duplicates in the products table.
WITH duplicate_in_products AS (
       SELECT *
       FROM `urbanloom_analytics.products`
       WHERE product_id IN (
          SELECT product_id
          FROM `urbanloom_analytics.products`
          GROUP BY product_id
          HAVING COUNT(*) > 1)
)
--Find out how many unique cost_price for each duplicate product_id.
SELECT
       product_id,
       COUNT(DISTINCT cost_price) AS unique_cost_count
FROM duplicate_in_products
GROUP BY product_id -- all unique_cost_count = 1, great news! -- If
unique_cost_count = 1 for all product_ids: All duplicates have identical
cost_price.
-- This confirms it's safe to remove duplicates after JOIN without affecting
calculations.
-- Any product_id with unique_cost_count > 1 would require further investigation.


--VALIDATION 2: Examine all fields in the duplicate records.
SELECT *
FROM `urbanloom_analytics.order_transactions`
WHERE order_transaction_id IN (
       SELECT order_transaction_id
       FROM `urbanloom_analytics.order_transactions`
       GROUP BY order_transaction_id
       HAVING COUNT(*) > 1)
--OBSERVATION: The order_transactions table contains no fields directly related to
cost_price, quantity, or revenue calculations.
--CONCLUSION: Safe to deduplicate after JOIN without impacting financial accuracy.
```

INSIGHT 2: Size preference and stock inventory for the next half year of 2024

1. Extended sizes (XS, XL, and sizes like 1X-4X) consistently have lower quantities, suggesting either lower demand or different ordering strategies.
2. Based on this inventory data from the first 6 months of performance, we can make predictions for future stock requirements. As shown in Table 1, June revenue roughly doubled that of January, indicating consistent growth momentum. To adequately support this anticipated business trajectory and meet rising customer demand over the next 6 months, the total inventory quantities for each category and size will need to be increased by approximately 2-3 times the current levels shown in this table.

product_category	total_quantity_XS	total_quantity_Small	total_quantity_Medium	total_quantity_Large	total_quantity_XL	total_quantity_1X	total_quantity_2X	total_quantity_3X	total_quantity_4X
Bottoms	1	11	18	23	17	2	1	2	
Jeans		4	9	7	2	2	3	2	
Loungewear	6	77	160	177	88	45	14	15	
Outerwear	9	25	101	131	104	62	35	16	2
Shirts	25	314	1206	1618	852	633	215	104	
Sportswear	39	168	231	216	154	114	84	51	
Tops	7	42	92	63	75	39	10	10	1

Table 2. Inventory Distribution by Product Category and Size. Current levels across 7 apparel categories and 9 size variants, with color coding indicating sales performance levels (green = highest sales volumes, red = lowest sales volumes). The screenshot of the query result was also attached below.

```
/*
 INSIGHT 2: Sales quantity analysis by product category and size for inventory planning
 STEP 1: Cleaned duplicate records using row_number() to ensure accurate quantity counts
 STEP 2: Created a pivot table showing total quantities sold by category and size
 STEP 3: Ordered by product_category for easy reference by the inventory team
*/
WITH order_product_row_number AS (
        SELECT
                ROW_NUMBER() OVER (PARTITION BY oli.transaction_line_item_id ORDER BY
                oli.transaction_line_item_id) AS row_number,
                *
        FROM `urbanloom_analytics.order_line_items` oli
        LEFT JOIN `urbanloom_analytics.products` p
                ON oli.product_id = p.product_id
        LEFT JOIN `urbanloom_analytics.order_transactions` ot
                ON oli.order_transaction_id = ot.order_transaction_id
        WHERE EXTRACT(MONTH FROM DATE(ot.order_transaction_date)) < 7
),
clean_order_product AS (
SELECT *
FROM order_product_row_number
WHERE row_number = 1
),--STEP 1
category_size_quantity AS(
        SELECT
                product_category,
```

```
                product_size,
                Quantity
        FROM clean_order_product
        WHERE product_category IS NOT NULL
), --STEP 2 (1)

pivot_table AS (
         SELECT
                *
        FROM category_size_quantity
        PIVOT (SUM(quantity) AS total_quantity for product_size IN ('XS','Small','Medium', 'Large',
        'XL', '1X','2X','3X','4X'))
) -- STEP 2(2)

SELECT *
FROM pivot_table
ORDER BY product_category -- STEP 3
```

Row	product_category	total_quantity_XS	total_quantity_Small	total_quantity_Medium	total_quantity_Large	total_quantity_XL	total_quantity_1X	total_quantity_2X	total_quantity_3X	total_quantity_4X
1	Bottoms	1	11	18	23	17	2	1	2	null
2	Jeans	null	4	9	7	2	2	3	2	null
3	Loungewear	6	77	160	177	88	45	14	15	null
4	Outerwear	9	25	101	131	104	62	35	16	2
5	Shirts	25	314	1206	1618	852	633	215	104	null
6	Sportswear	39	168	231	216	154	114	84	51	null
7	Tops	7	42	92	63	75	39	10	10	1

Screenshot of query results

INSIGHT 3: Top 3 potential locations for the new physical store: London, Manchester and Sheffield

1. Highest Revenue Performance: These three cities generated the highest total revenue over the first 6 months, demonstrating strong market demand for Urban Loom's products.
2. Consistent Monthly Growth: All three locations show an increasing performance trend, indicating positive market momentum and growing brand recognition.

Row	city	month	total_revenue
1	*null*	*null*	499263.66
2	London	*null*	33836.09
3	London	1	3260.17
4	London	2	3211.72
5	London	3	5493.79
6	London	4	8482.32
7	London	5	5470.02
8	London	6	7918.07
9	Manchester	*null*	15388.84
10	Manchester	1	1800.97
11	Manchester	2	1357.16
12	Manchester	3	2329.3
13	Manchester	4	2849.63
14	Manchester	5	2065.15
15	Manchester	6	4986.63
16	Sheffield	*null*	15167.34
17	Sheffield	1	1613.18
18	Sheffield	2	1600.38
19	Sheffield	3	2676.45
20	Sheffield	4	2404.09
21	Sheffield	5	3287.14
22	Sheffield	6	3586.1

Table 3. Revenue performance for the top 3 UK cities by total revenue, showing a monthly breakdown from January to June 2024.

```sql
/*
    INSIGHT 3: Urban Loom Store Location Analysis
    STEP 1: Cleaned the duplicates and focused only on cities in the UK and excluded July (Month 7)
from analysis
    STEP 2: Used ROLLUP to show both city revenue totals and breakdowns by city and month
    STEP 3: Ordered the results by total revenue and identified the top 3 potential locations for the new
store.
*/
WITH customer_items_transaction AS (
        SELECT
                c.city,
                c.country,
                ot.order_transaction_date,
                oli.*,
                ROW_NUMBER() OVER (PARTITION BY oli.transaction_line_item_id ORDER BY
        oli.transaction_line_item_id) AS row_number
        FROM `urbanloom_analytics.order_line_items` oli
        LEFT JOIN `urbanloom_analytics.order_transactions` ot
                ON oli.order_transaction_id = ot.order_transaction_id
        LEFT JOIN `urbanloom_analytics.customers` c
                ON ot.customer_id = c.customer_id
),

cleaned_customer_items_transaction AS (
         SELECT *
         FROM customer_items_transaction
         WHERE row_number = 1 -- STEP 1 (1)
)

SELECT
city,
EXTRACT(MONTH FROM DATE(order_transaction_date)) AS month,
ROUND(SUM(revenue), 2) AS total_revenue,
FROM cleaned_customer_items_transaction
WHERE country = 'United Kingdom'  -- STEP 1 (2)
AND EXTRACT(MONTH FROM DATE(order_transaction_date)) < 7 -- STEP 1 (3)
GROUP BY ROLLUP (city, month) -- STEP 2
ORDER BY
SUM(SUM(revenue)) OVER (PARTITION BY city) DESC,-- STEP 3 (1)
city,
month
LIMIT 22 -- STEP 3 (2)
```