SQL

DATA ANALYSIS PROJECT

8WEEKSQLCHALLENGE.COM

CASE STUDY #6

CliqueBait

ATTENTION CAPTURING

DATAWITHDANNY.COM

Shaik Shoaib Akthar

Introduction

Clique Bait is not like your regular online seafood store - the founder and CEO Danny, was also a part of a digital data analytics team and wanted to expand his knowledge into the seafood industry!

In this case study - you are required to support Danny's vision and analyze his dataset and come up with creative solutions to calculate funnel fallout rates for the Clique Bait online store.

Table 1: Users

Customers who visit the Clique Bait website are tagged via their **cookie_id**

user_id	cookie_id	start_date
397	3759ff	2020-03-30 00:00:00
215	863329	2020-01-26 00:00:00
191	eefca9	2020-03-15 00:00:00
89	764796	2020-01-07 00:00:00
127	17ccc5	2020-01-22 00:00:00
81	b0b666	2020-03-01 00:00:00
260	a4f236	2020-01-08 00:00:00
203	d1182f	2020-04-18 00:00:00
23	12dbc8	2020-01-18 00:00:00
375	f61d69	2020-01-03 00:00:00

Table 2: Events

Customer visits are logged in this **events** table at a **cookie_id** level and the **event_type** and **page_id** values can be used to join onto relevant satellite tables to obtain further information about each event.

The **sequence_number** is used to order the events within each visit.

visit_id	cookie_id	page_id	event_type	sequence_number	event_time
719fd3	3d83d3	5	1	4	2020-03-02 00:
fb1eb1	c5ff25	5	2	8	2020-01-22 07:
23fe81	1e8c2d	10	1	9	2020-03-21 13:
ad91aa	648115	6	1	3	2020-04-27 16:
5576d7	ac418c	6	1	4	2020-01-18 04:
48308b	c686c1	8	1	5	2020-01-29 06:
46b17d	78f9b3	7	1	12	2020-02-16 09:
9fd196	ccf057	4	1	5	2020-02-14 08:
edf853	f85454	1	1	1	2020-02-22 12:
3c6716	02e74f	3	2	5	2020-01-31 17:

Table 3: Event Identifier

The **event_identifier** table shows the types of events which are captured by Clique Bait's digital data systems.

event_type	event_name
1	Page View
2	Add to Cart
3	Purchase
4	Ad Impression
5	Ad Click

Table 4: Campaign Identifier

This table shows information for the 3 campaigns that Clique Bait has ran on their website so far in 2020.

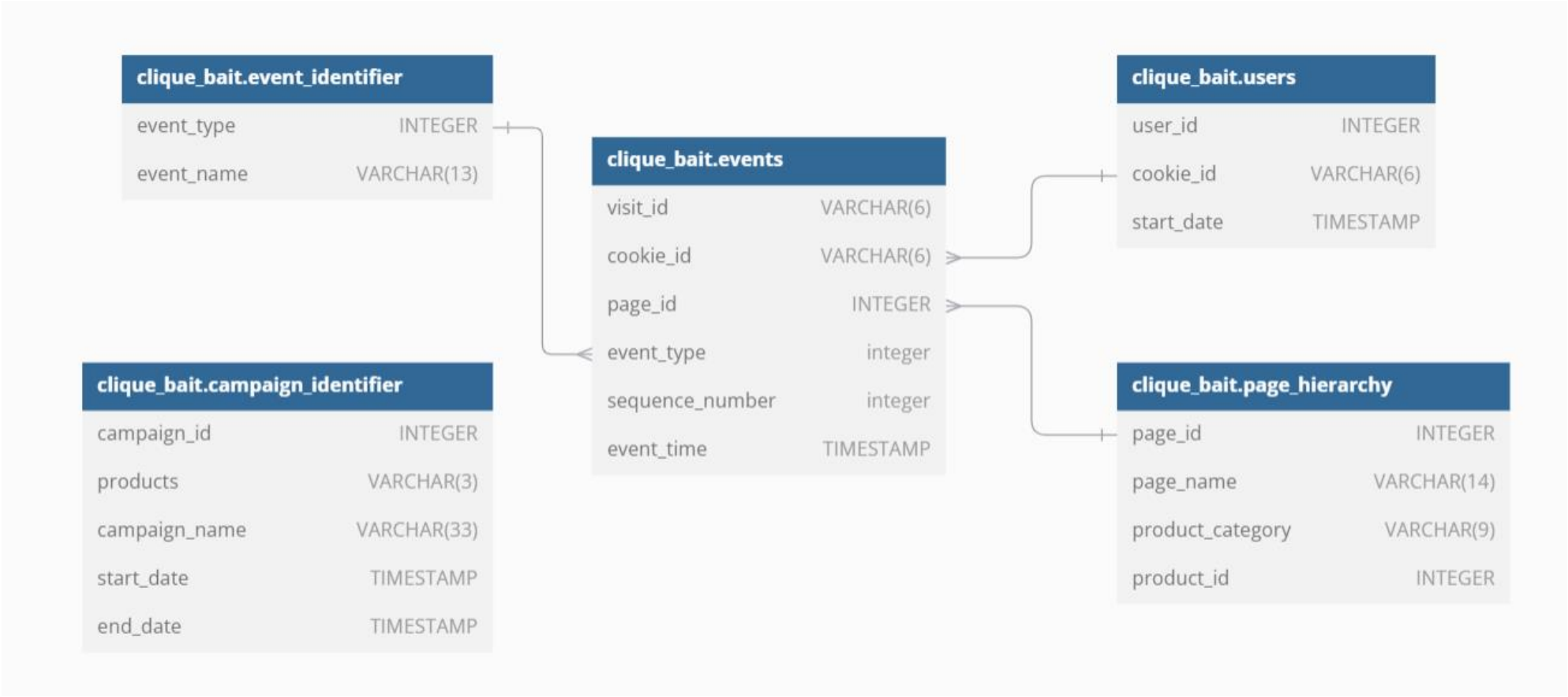
campaign_id	products	campaign_name	start_date	end_date
1	1-3	BOGOF - Fishing For Compliments	2020-01-01 00:00:00	2020-01-14 00:00:00
2	4-5	25% Off - Living The Lux Life	2020-01-15 00:00:00	2020-01-28 00:00:00
3	6-8	Half Off - Treat Your Shellf(ish)	2020-02-01 00:00:00	2020-03-31 00:00:00

Table 5: Page Hierarchy

This table lists all of the pages on the Clique Bait website which are tagged and have data passing through from user interaction events.

page_id	page_name	product_category	product_id
1	Home Page	null	null
2	All Products	null	null
3	Salmon	Fish	1
4	Kingfish	Fish	2
5	Tuna	Fish	3
6	Russian Caviar	Luxury	4
7	Black Truffle	Luxury	5
8	Abalone	Shellfish	6
9	Lobster	Shellfish	7
10	Crab	Shellfish	8
11	Oyster	Shellfish	9
12	Checkout	null	null
13	Confirmation	null	null

Entity Relationship Diagram



Digital Analysis

```sql
-- 1. How many users are there?
SELECT COUNT(DISTINCT user_id) AS user_count
FROM users;
```

user_count
500

```sql
-- 2. How many cookies does each user have on average?
WITH cookies AS (
  SELECT user_id, COUNT(cookie_id) AS cookie_count
  FROM users
  GROUP BY user_id
)
SELECT AVG(cookie_count) as avg_cookie_count
FROM cookies;
```

avg_cookie_count
3.5640

```sql
-- 3. What is the unique number of visits by all users per month?
SELECT
  MONTH(event_time) AS months,
  COUNT(DISTINCT visit_id) AS visits_count
FROM events
GROUP BY months
ORDER BY months;
```

months	visits_count
1	876
2	1488
3	916
4	248
5	36

```sql
-- 4. What is the number of events for each event type?
SELECT
  e.event_type,
  ei.event_name,
  COUNT(*) AS event_count
FROM events e
LEFT JOIN event_identifier ei ON e.event_type = ei.event_type
GROUP BY e.event_type, ei.event_name
ORDER BY e.event_type;
```

event_type	event_name	event_count
1	Page View	20928
2	Add to Cart	8451
3	Purchase	1777
4	Ad Impression	876
5	Ad Click	702

```sql
-- 5. What is the percentage of visits which have a purchase event?
SELECT
  ROUND((COUNT(DISTINCT e.visit_id)/
  (SELECT COUNT(DISTINCT visit_id) FROM events))*100, 2) AS purchase_pct
FROM events e
LEFT JOIN event_identifier ei ON e.event_type = ei.event_type
WHERE ei.event_name = 'Purchase';
```

purchase_pct
49.86

```sql
-- 6. What is the percentage of visits which view the checkout page but do not have
a purchase event?
-- count of visits which has checkout page but not purchase event/count of visits
which has checkout page
SELECT
  ROUND(
    ((SELECT COUNT(DISTINCT visit_id)
    FROM events
    WHERE page_id = 12 AND
    visit_id NOT IN (SELECT DISTINCT visit_id FROM events WHERE event_type = 3))/
    (SELECT COUNT(DISTINCT visit_id)
     FROM events
     WHERE page_id = 12))
  *100, 2)
AS view_checkout_no_purchase_pct;
```

view_checkout_no_purchase_pct
15.50

```sql
-- 7. What are the top 3 pages by number of views?
SELECT p.page_name,
       COUNT(*) AS view_count
FROM events e
LEFT JOIN page_hierarchy p ON e.page_id = p.page_id
WHERE e.event_type = 1
GROUP BY p.page_name
ORDER BY view_count DESC
LIMIT 3;
```

page_name	view_count
All Products	3174
Checkout	2103
Home Page	1782

```sql
-- 8. What is the number of views and cart adds for each product category?
SELECT
  p.product_category,
  SUM(CASE WHEN ei.event_name = 'Page View' THEN 1 ELSE 0 END) AS page_views,
  SUM(CASE WHEN ei.event_name = 'Add to Cart' THEN 1 ELSE 0 END) AS cart_adds
FROM events e
JOIN event_identifier ei ON e.event_type = ei.event_type
JOIN page_hierarchy p ON e.page_id = p.page_id
WHERE p.product_category IS NOT NULL
GROUP BY p.product_category
ORDER BY page_views DESC;
```

product_category	page_views	cart_adds
Shellfish	6204	3792
Fish	4633	2789
Luxury	3032	1870

```sql
-- 9. What are the top 3 products by purchases?
SELECT p.product_id,
       p.page_name AS product_name,
       p.product_category,
       COUNT(*) AS purchase_count
FROM events e
JOIN event_identifier ei ON e.event_type = ei.event_type
JOIN page_hierarchy p ON e.page_id = p.page_id
WHERE ei.event_name = 'Add to Cart'
AND e.visit_id IN (
  SELECT e.visit_id
  FROM events e
  JOIN event_identifier ei ON e.event_type = ei.event_type
  WHERE ei.event_name = 'Purchase')
GROUP BY p.product_id, product_name, p.product_category
ORDER BY purchase_count DESC
LIMIT 3;
```

product_id	product_name	product_category	purchase_count
7	Lobster	Shellfish	754
9	Oyster	Shellfish	726
8	Crab	Shellfish	719

Product Funnel Analysis

```sql
CREATE TEMPORARY TABLE product_summary AS
-- How many times was each product viewed?
WITH product_views AS (
SELECT p.product_id, p.page_name, COUNT(*) AS page_views
FROM events e
JOIN page_hierarchy p ON e.page_id = p.page_id
WHERE e.event_type = 1 AND p.product_id IS NOT NULL
GROUP BY p.page_name, p.product_id
),

-- How many times was each product added to cart?
cart_adds AS (
SELECT p.page_name, COUNT(*) AS add_cart_count
FROM events e
JOIN page_hierarchy p ON e.page_id = p.page_id
WHERE e.event_type = 2 AND p.product_id IS NOT NULL
GROUP BY p.page_name
),

-- How many times was each product added to a cart but not purchased (abandoned)?
added_cart_not_purchased AS (
SELECT p.page_name,
  COUNT(*) AS added_cart_no_purchase_count
FROM events e
JOIN page_hierarchy p ON e.page_id = p.page_id
WHERE e.event_type = 2 AND e.visit_id NOT IN (
  SELECT e.visit_id
  FROM events e
  WHERE e.event_type = 3)
GROUP BY p.page_name
),

-- How many times was each product purchased?
purchases AS (
SELECT p.page_name, COUNT(*) AS purchase_count
FROM events e
JOIN page_hierarchy p ON e.page_id = p.page_id
JOIN event_identifier ei ON e.event_type = ei.event_type
WHERE ei.event_name = 'Add to Cart'
AND e.visit_id IN ( SELECT e.visit_id
  FROM events e
  JOIN event_identifier ei ON e.event_type = ei.event_type
  WHERE ei.event_name = 'Purchase' )
GROUP BY p.page_name
)

SELECT pv.*,
  ca.add_cart_count,
    anp.added_cart_no_purchase_count,
    p.purchase_count
FROM product_views pv
JOIN cart_adds ca ON pv.page_name = ca.page_name
JOIN added_cart_not_purchased anp ON pv.page_name = anp.page_name
JOIN purchases p ON pv.page_name = p.page_name
ORDER BY pv.product_id;

-- check the created table
SELECT * FROM product_summary;
```

product_id	page_name	page_views	add_cart_count	added_cart_no_purchase_count	purchase_count
1	Salmon	1559	938	227	711
2	Kingfish	1559	920	213	707
3	Tuna	1515	931	234	697
4	Russian Caviar	1563	946	249	697
5	Black Truffle	1469	924	217	707
6	Abalone	1525	932	233	699
7	Lobster	1547	968	214	754
8	Crab	1564	949	230	719
9	Oyster	1568	943	217	726

```sql
-- How many times was each product viewed?
WITH product_views AS (
SELECT p.product_category, COUNT(*) AS page_views
FROM events e
JOIN page_hierarchy p ON e.page_id = p.page_id
WHERE e.event_type = 1 AND p.product_id IS NOT NULL
GROUP BY p.product_category
),

-- How many times was each product added to cart?
cart_adds AS (
SELECT p.product_category, COUNT(*) AS add_cart_count
FROM events e
JOIN page_hierarchy p ON e.page_id = p.page_id
WHERE e.event_type = 2 AND p.product_id IS NOT NULL
GROUP BY p.product_category
),

-- How many times was each product added to a cart but not purchased (abandoned)?
added_cart_not_purchased AS (
SELECT p.product_category,
     COUNT(*) AS added_cart_no_purchase_count
FROM events e
JOIN page_hierarchy p ON e.page_id = p.page_id
WHERE e.event_type = 2 AND e.visit_id NOT IN (
  SELECT e.visit_id
  FROM events e
  WHERE e.event_type = 3)
GROUP BY p.product_category
),

-- How many times was each product purchased?
purchases AS (
SELECT p.product_category, COUNT(*) AS purchase_count
FROM events e
JOIN page_hierarchy p ON e.page_id = p.page_id
JOIN event_identifier ei ON e.event_type = ei.event_type
WHERE ei.event_name = 'Add to Cart'
AND e.visit_id IN (
  SELECT e.visit_id
  FROM events e
  JOIN event_identifier ei ON e.event_type = ei.event_type
  WHERE ei.event_name = 'Purchase' )
GROUP BY p.product_category
),

category_summary AS (
SELECT pv.*,
  ca.add_cart_count,
  anp.added_cart_no_purchase_count,
  p.purchase_count
FROM product_views pv
JOIN cart_adds ca ON pv.product_category = ca.product_category
JOIN added_cart_not_purchased anp ON pv.product_category = anp.product_category
JOIN purchases p ON pv.product_category = p.product_category
ORDER BY pv.product_category
)

-- check the category summary
SELECT * FROM category_summary;
```

product_category	page_views	add_cart_count	added_cart_no_purchase_count	purchase_count
Fish	4633	2789	674	2115
Luxury	3032	1870	466	1404
Shellfish	6204	3792	894	2898

```sql
-- product with most views
SELECT *
FROM product_summary
ORDER BY page_views DESC
LIMIT 1;
```

product_id	page_name	page_views	add_cart_count	added_cart_no_purchase_count	purchase_count
9	Oyster	1568	943	217	726

```sql
-- product with most cart adds
SELECT *
FROM product_summary
ORDER BY add_cart_count DESC
LIMIT 1;
```

product_id	page_name	page_views	add_cart_count	added_cart_no_purchase_count	purchase_count
7	Lobster	1547	968	214	754

```sql
-- product with most purchases
SELECT *
FROM product_summary
ORDER BY purchase_count DESC
LIMIT 1;
```

product_id	page_name	page_views	add_cart_count	added_cart_no_purchase_count	purchase_count
7	Lobster	1547	968	214	754

```sql
-- 2. Which product was most likely to be abandoned?
SELECT *
FROM product_summary
ORDER BY added_cart_no_purchase_count DESC
LIMIT 1;
```

product_id	page_name	page_views	add_cart_count	added_cart_no_purchase_count	purchase_count
4	Russian Caviar	1563	946	249	697

```sql
-- 3. Which product had the highest view to purchase percentage?
SELECT product_id,
       page_name,
       ROUND((purchase_count/page_views)*100, 2) AS purchase_view_pct
FROM product_summary
ORDER BY purchase_view_pct DESC
LIMIT 1;
```

product_id	page_name	purchase_view_pct
7	Lobster	48.74

```sql
-- 4. What is the average conversion rate from view to cart add?
SELECT ROUND(AVG(add_cart_count/page_views)*100, 2) AS avg_view_cart_conv_rate
FROM product_summary
ORDER BY avg_view_cart_conv_rate DESC
LIMIT 1;
```

avg_view_cart_conv_rate
60.95

```sql
-- 5. What is the average conversion rate from cart add to purchase?
SELECT ROUND(AVG(purchase_count/add_cart_count)*100, 2) AS avg_cart_purchase_conv_rate
FROM product_summary
ORDER BY avg_cart_purchase_conv_rate DESC
LIMIT 1;
```

avg_cart_purchase_conv_rate
75.93

Campaigns Analysis

```sql
CREATE TEMPORARY TABLE campaign_summary AS
SELECT
    u.user_id,
    e.visit_id,
    MIN(event_time) AS visit_start_time,
    SUM(CASE WHEN ei.event_name = 'Page View' THEN 1 ELSE 0 END) AS page_views,
    SUM(CASE WHEN ei.event_name = 'Add to Cart' THEN 1 ELSE 0 END) AS cart_adds,
    SUM(CASE WHEN ei.event_name = 'Purchase' THEN 1 ELSE 0 END) AS purchase,
    c.campaign_name,
    SUM(CASE WHEN ei.event_name = 'Ad Impression' THEN 1 ELSE 0 END) AS impression,
    SUM(CASE WHEN ei.event_name = 'Ad Click' THEN 1 ELSE 0 END) AS click,
    GROUP_CONCAT(CASE WHEN ei.event_name = 'Add to Cart' THEN ph.page_name END
        ORDER BY e.sequence_number SEPARATOR ', ') AS cart_products
FROM events e
JOIN users u ON e.cookie_id = u.cookie_id
JOIN event_identifier ei ON e.event_type = ei.event_type
JOIN page_hierarchy ph ON e.page_id = ph.page_id
LEFT JOIN campaign_identifier c ON e.event_time BETWEEN c.start_date AND c.end_date
GROUP BY u.user_id, e.visit_id, c.campaign_name;

SELECT * FROM campaign_summary
LIMIT 5;
```

user_id	visit_id	visit_start_time	page_views	cart_adds	purchase	campaign_name	impression	click	cart_products
1	02a5d5	2020-02-26 16:57:26	4	0	0	Half Off - Treat Your Shellf(ish)	0	0	NULL
1	0826dc	2020-02-26 05:58:38	1	0	0	Half Off - Treat Your Shellf(ish)	0	0	NULL
1	0fc437	2020-02-04 17:49:50	10	6	1	Half Off - Treat Your Shellf(ish)	1	1	Tuna, Russian Caviar, Black Truffle, Abalone, Crab, Oyster
1	30b94d	2020-03-15 13:12:54	9	7	1	Half Off - Treat Your Shellf(ish)	1	1	Salmon, Kingfish, Tuna, Russian Caviar, Abalone, Lobster, Crab
1	41355d	2020-03-25 00:11:18	6	1	0	Half Off - Treat Your Shellf(ish)	0	0	Lobster

```sql
-- Calculate no. of users who received impressions during campaign period
SELECT COUNT(DISTINCT user_id) AS received_impressions
FROM campaign_summary
WHERE impression > 0
AND campaign_name IS NOT NULL;
```

received_impressions
417

```sql
-- Calculate no.of users who received impressions but didn't click on ad
CREATE TEMPORARY TABLE temp_received_clicked AS
  SELECT DISTINCT user_id
  FROM campaign_summary
  WHERE campaign_name IS NOT NULL
  AND click > 0;

SELECT COUNT(DISTINCT user_id) AS received_impressions_no_click
FROM campaign_summary
WHERE impression > 0
AND campaign_name IS NOT NULL
AND user_id NOT IN (
SELECT user_id FROM temp_received_clicked
);
```

received_impressions_no_click
50

```sql
-- Calculate no.of users who received impressions and clicked on ad
SELECT COUNT(DISTINCT user_id) AS received_impressions_clicked
FROM campaign_summary
WHERE impression > 0
AND campaign_name IS NOT NULL
AND user_id IN (
SELECT user_id FROM temp_received_clicked
);
```

received_impressions_clicked
367

```sql
-- Calculate no. of users who didn't receive impressions
CREATE TEMPORARY TABLE temp_users_impressions AS
  SELECT DISTINCT user_id
  FROM campaign_summary
  WHERE campaign_name IS NOT NULL
  AND impression > 0;

SELECT COUNT(DISTINCT user_id) AS no_impressions
FROM campaign_summary
WHERE campaign_name IS NOT NULL
AND user_id NOT IN (SELECT user_id FROM temp_users_impressions);
```

no_impressions
78

Observations

- No.of users who received impressions during campaign period is 417
- Among these users who received impressions, 50 users didn't click on the ad
- No. of uses who received impressions and clicked on the ad is 367
- No. of users who didn't receive impressions during campaign period is 78
- Impression rate = No.of users who received impressions/Total users in the campaign period = (417/(417 + 78))*100 = 84.24%
- Ad-click rate = No.of users who clicked the ad/No. of users who received impressions = (367/417)*100 = 88.01%

```sql
-- Calculate average views, cart adds, purchases for each group

-- Users who received impressions
SET @received = 417;

SELECT CAST(SUM(page_views)/@received AS DECIMAL(10, 1)) AS avg_page_views,
    CAST(SUM(cart_adds)/@received AS DECIMAL(10,1)) AS avg_cart_adds,
    CAST(SUM(purchase)/@received AS DECIMAL(10,1)) AS avg_purchase
FROM campaign_summary
WHERE impression > 0
AND campaign_name IS NOT NULL;
```

avg_page_views	avg_cart_adds	avg_purchase
15.3	9.0	1.5

```sql
-- Users who received impressions and didn't click on ad
SET @received_no_click = 50;

SELECT CAST(SUM(page_views)/@received_no_click AS DECIMAL(10, 1)) AS avg_page_views,
    CAST(SUM(cart_adds)/@received_no_click AS DECIMAL(10,1)) AS avg_cart_adds,
    CAST(SUM(purchase)/@received_no_click AS DECIMAL(10,1)) AS avg_purchase
FROM campaign_summary
WHERE impression > 0
AND campaign_name IS NOT NULL
AND user_id NOT IN (
SELECT user_id FROM temp_received_clicked
);
```

avg_page_views	avg_cart_adds	avg_purchase
7.6	2.7	0.8

```sql
-- Users who received impressions and clicked on ad
SET @received_clicked = 367;

SELECT CAST(SUM(page_views)/@received_clicked AS DECIMAL(10, 1)) AS avg_page_views,
    CAST(SUM(cart_adds)/@received_clicked AS DECIMAL(10,1)) AS avg_cart_adds,
    CAST(SUM(purchase)/@received_clicked AS DECIMAL(10,1)) AS avg_purchase
FROM campaign_summary
WHERE impression > 0
AND campaign_name IS NOT NULL
AND user_id IN (
SELECT user_id FROM temp_received_clicked
);
```

avg_page_views	avg_cart_adds	avg_purchase
16.4	9.9	1.6

```sql
-- Users who didn't receive impressions
SET @not_received = 78;

SELECT CAST(SUM(page_views)/@not_received AS DECIMAL(10, 1)) AS avg_page_views,
    CAST(SUM(cart_adds)/@not_received AS DECIMAL(10,1)) AS avg_cart_adds,
    CAST(SUM(purchase)/@not_received AS DECIMAL(10,1)) AS avg_purchase
FROM campaign_summary
WHERE campaign_name IS NOT NULL
AND user_id NOT IN (SELECT user_id FROM temp_users_impressions);
```

avg_page_views	avg_cart_adds	avg_purchase
19.0	5.6	1.3

Insights

- Average purchase of users who received impressions is 1.5
- Average purchase of users who received impressions but didn't click on ad is 0.8
- Average purchase of users who received impressions and clicked on ad is 1.6
- Average purchase of users who didn't receive impressions is 1.3
- Clicking on impressions lead to higher purchase rate (1.6 > 1.3)

Insights

Uplift in purchase rate when comparing users who click on a campaign impression vs. users who do not receive an impression

= ((1.6 - 1.3)/1.3)*100 = 23.08%

If we compare them with users who just got impression but do not click

= ((1.6 - 0.8)/0.8)*100 = 100%

- Clicking on the ad leads to a 23.08% higher purchase rate compared to those who did not receive an impression.
- Clicking on the ad leads to a 100% higher purchase rate compared to those who received an impression but didn't click.

```sql
-- average views, cart adds and purchases for each campaign
-- for users who received impressions
SELECT campaign_name,
  CAST(SUM(page_views)/@received AS DECIMAL(10, 1)) AS avg_page_views,
    CAST(SUM(cart_adds)/@received AS DECIMAL(10,1)) AS avg_cart_adds,
    CAST(SUM(purchase)/@received AS DECIMAL(10,1)) AS avg_purchase
FROM campaign_summary
WHERE impression > 0
AND campaign_name IS NOT NULL
GROUP BY campaign_name;
```

campaign_name	avg_page_views	avg_cart_adds	avg_purchase
Half Off - Treat Your Shellf(ish)	11.8	6.9	1.2
25% Off - Living The Lux Life	2.2	1.3	0.2
BOGOF - Fishing For Compliments	1.4	0.8	0.1

The "Half Off - Treat Your Shellf(ish)" campaign performed the best across all metrics - highest page views, cart adds, and purchases - indicating that it was the most successful in driving both interest and conversions.

```sql
-- Conversion Rate for each campaign
-- ratio of no.of purchases to the no.of impressions
SELECT campaign_name,
  SUM(purchase) AS purchases,
    SUM(impression) AS impressions,
  CAST(SUM(purchase)/SUM(impression) AS DECIMAL(10, 2)) AS conversion_rate
FROM campaign_summary
WHERE impression > 0
AND campaign_name IS NOT NULL
GROUP BY campaign_name;
```

campaign_name	purchases	impressions	conversion_rate
Half Off - Treat Your Shellf(ish)	493	578	0.85
25% Off - Living The Lux Life	87	104	0.84
BOGOF - Fishing For Compliments	55	65	0.85

Insights & Recommendations

- "Half Off - Treat Your Shellf(ish)" is the standout campaign in terms of total sales and reach, despite having a conversion rate similar to the other two campaigns.
- The high conversion rates across all campaigns suggest that users who engage are highly likely to purchase, but the total impressions (reach) play a major role in determining overall success.
- To improve the impact of campaigns like "Living The Lux Life" and "Fishing For Compliments", focusing on increasing impressions (i.e., expanding the audience) could lead to higher total purchases.

```sql
-- Click Through Rate for each campaign
-- ratio of no.of clicks to the no.of impressions
SELECT campaign_name,
  SUM(click) AS clicks,
    SUM(impression) AS impressions,
  CAST(SUM(click)/SUM(impression) AS DECIMAL(10, 2)) AS click_through_rate
FROM campaign_summary
WHERE impression > 0
AND campaign_name IS NOT NULL
GROUP BY campaign_name;
```

campaign_name	clicks	impressions	click_through_rate
Half Off - Treat Your Shellf(ish)	463	578	0.80
25% Off - Living The Lux Life	81	104	0.78
BOGOF - Fishing For Compliments	55	65	0.85

Insights & Recommendations

- "BOGOF - Fishing For Compliments" is the most engaging campaign in terms of CTR, but due to the limited reach (fewer impressions), it generated fewer total clicks.
- "Half Off - Treat Your Shellf(ish)" strikes a good balance between high CTR and larger reach, making it the overall winner in terms of driving clicks and engagement.
- All campaigns have strong click-through rates, but increasing impressions for the campaigns with fewer clicks could improve their overall performance.

THANK YOU