PROJECT TITLE: NIK Fuzzy Factory eCommerce Database Analysis

Overview:

Welcome to a groundbreaking project that goes beyond the conventional realm of SQL practice. In the NIK Fuzzy Factory eCommerce Database Analysis project, you will delve into a meticulously crafted eCommerce database tailored for real-world applications. Unlike generic examples, this project immerses you in the role of a Database Analyst at NIK Fuzzy Factory, an innovative eCommerce start-up.

Project Objectives:

As a Database Analyst for NIK Fuzzy Factory, you will collaborate directly with the CEO, Marketing Director, and Website Manager. Your primary goal is to leverage advanced SQL tools and techniques to analyze and enhance business performance.

Key Areas of Analysis:

1. **Analyzing Traffic Sources:**
    - Identify Top Traffic Sources
    - Analyze Traffic Conversion Rates
    - Explore Traffic Source Trending
    - Optimize Traffic Source Bids
    - Segment Traffic Source Trending
2. **Analyzing Website Performance:**
    - Identify Top Website Pages
    - Determine Top Entry Pages
    - Calculate Bounce Rates
    - Analyze Landing Page Tests
    - Conduct Landing Page Trend Analysis
    - Build Conversion Funnels
    - Analyze Conversion Funnel Tests
3. **Channel Management Analysis:**
    - Analyze Channel Portfolios
    - Compare Channel Characteristics
    - Optimize Bids Across Channels
    - Explore Channel Portfolio Trends
    - Analyze Free Channels
4. **Business Patterns & Seasonality Analysis:**

    - Analyze Seasonality Trends
    - Identify Business Patterns
    - Uncover Seasonal Business Patterns
5. **Product Analysis:**
    - Conduct Product Level Sales Analysis
    - Analyze Product Launch Sales
    - Examine Product Pathing
    - Build Product Conversion Funnels
    - Cross-Sell Analysis
    - Portfolio Expansion Analysis
    - Analyze Product Refund Rates
6. **User Analysis:**
    - Identify Repeat Visitors
    - Analyze Repeat Behavior
    - Explore New vs. Repeat Channel Patterns
    - Evaluate New vs. Repeat Performance

Benefits:

- Gain hands-on experience with a custom eCommerce database.
- Apply SQL skills to real-world scenarios.
- Work closely with industry professionals to grow an eCommerce business.
- Acquire a deep understanding of data analysis in an eCommerce environment.

This project is a unique opportunity to hone your analytical skills, work on real projects, and truly think like an analyst in a dynamic eCommerce setting. Get ready to revolutionize your SQL capabilities and make a tangible impact on NIK Fuzzy Factory's success.

About The Dataset:

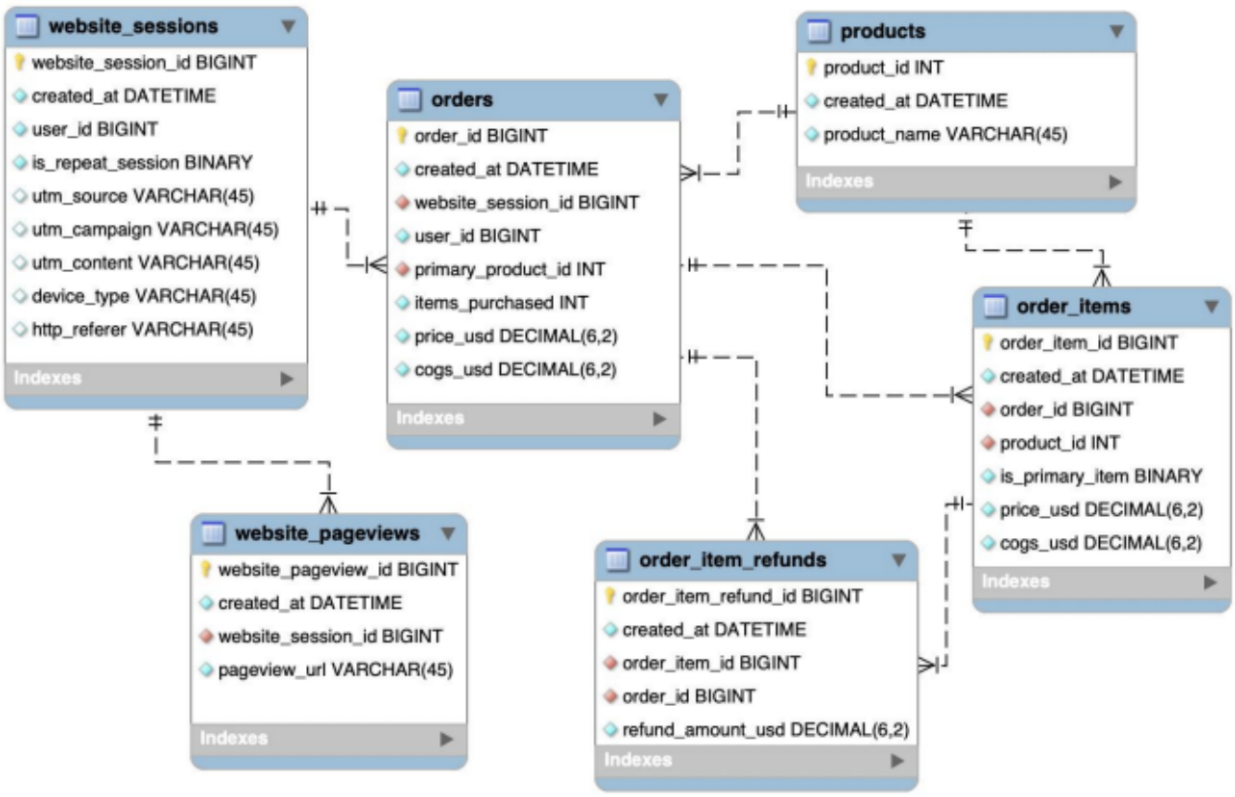
We will be working with six related tables, which contain eCommerce data about:

• Website Activity

• Products

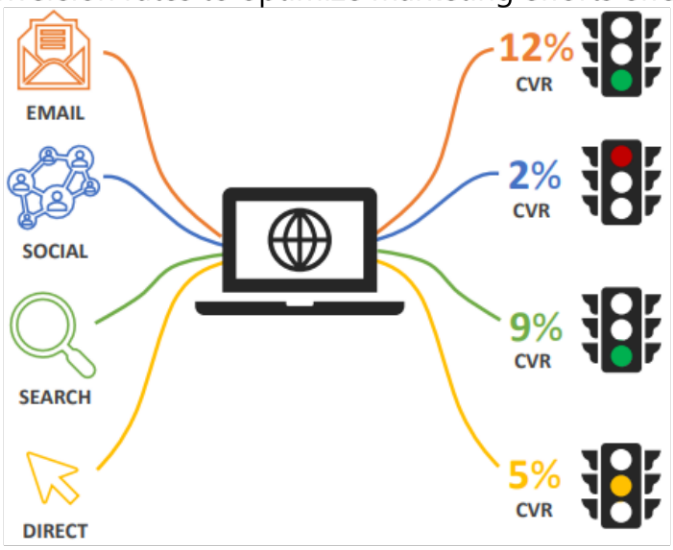
• Orders and Refunds

We'll use MySQL to understand how customers access and interact with the site, analyze landing page performance and conversion, and explore product-level sales.



Analyzing Traffic Sources:

Traffic source analysis is a vital component of marketing strategy, focusing on the comprehensive understanding of customer acquisition channels and the identification of high-quality traffic sources. This process involves analyzing various aspects of user engagement and conversion rates to optimize marketing efforts effectively.



COMMON USE CASES:

• Analyzing search data and shifting budget towards the engines, campaigns or keywords driving the strongest conversion rates

• Comparing user behavior patterns across traffic sources to inform creative and messaging strategy

• Identifying opportunities to eliminate wasted spend or scale high-converting traffic

About WEBSITE_SESSION table:

What is UTM?

When businesses run paid marketing campaigns, they often obsess over performance and measure everything; how much they spend, how well traffic converts to sales, etc.

Paid traffic is commonly tagged with tracking (UTM) parameters, which are appended to URLs and allow us to tie website activity back to specific traffic sources and campaigns.

• We use the utm parameters stored in the database to identify paid website sessions

• From our session data, we can link to our order data to understand how much revenue our paid campaigns are driving

FINDING TOP TRAFFIC SOURCES

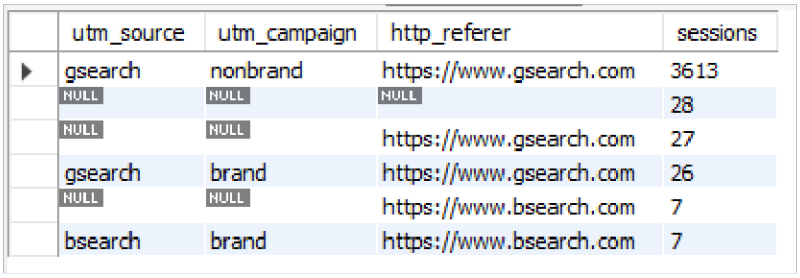
April 12, 2012

Over the past month, as you've been live and generating sales, could you provide insights into the primary sources of your website sessions? Specifically, I'd like a breakdown by UTM source, campaign, and referring domain through yesterday. This information would be valuable for understanding the distribution of traffic and optimizing our strategies.

SQL QUERY:

```
select
        utm_source,
   utm_campaign,
   http_referer,
   count(distinct website_session_id) as sessions
from website_sessions
where created_at < '2012-04-12'
group by
                utm_source,
                utm_campaign,
                http_referer
order by sessions desc;
```

Findings:



utm_source	utm_campaign	http_referer	sessions
gsearch	nonbrand	https://www.gsearch.com	3613
NULL	NULL	NULL	28
NULL	NULL	https://www.gsearch.com	27
gsearch	brand	https://www.gsearch.com	26
NULL	NULL	https://www.bsearch.com	7
bsearch	brand	https://www.bsearch.com	7

The findings suggest that further exploration and optimization efforts should be directed towards "gsearch nonbrand." Delving deeper into this area could uncover opportunities to enhance performance and refine strategies, potentially improving outcomes in terms of website sessions and conversions.

TRAFFIC CONVERSION RATES

April 14, 2012
Could you check how many people who visit our site through "gsearch nonbrand" end up making a purchase? We want to calculate the conversion rate, and if it's below 4%, we might need to spend less on clicks. On the flip side, if it's higher, we can consider investing more to attract even more visitors.

SQL QUERY:

```
select
```

```
        count(distinct website_sessions.website_session_id) as sessions,
    count(distinct orders.order_id) as orders,
    count(distinct orders.order_id) / count(distinct website_sessions.website_session_id) as
session_to_order_conv_rt
from website_sessions
        left join orders
                on orders.website_session_id = website_sessions.website_session_id
where website_sessions.created_at < '2012-04-14'
        and utm_source = 'gsearch'
  and utm_campaign = 'nonbrand';
```

Findings:

sessions	orders	session_to_order_conv_rt
3895	112	0.0288

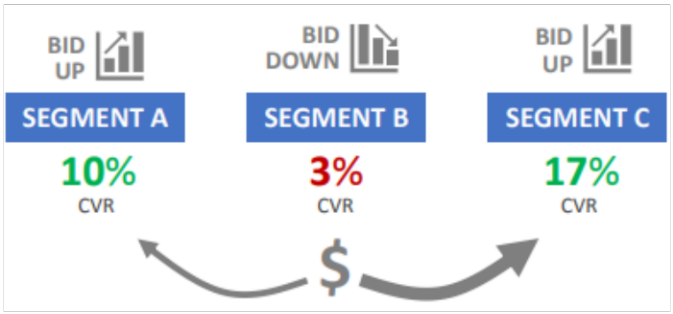
The analysis indicates that we're currently below the 4% threshold needed to sustain our economics. Consequently, it would be advisable to dial down our search bids. This adjustment is necessary to align our spending with the existing conversion rate and avoid over-spending on the current performance metrics.

What is Bid Optimization?

Analyzing bids for optimization is like figuring out the best way to spend your money on advertising. We look at how well different types of ads bring in customers and use data like conversion rates and revenue per click to decide how much money to spend on each click to get customers.

We also check how our website and products perform for different groups of people, like those using mobile or desktop. This helps us spend money where it works best.

And when we make changes to how much we're willing to pay for ads, we look at how it affects our position in ad auctions and how many customers come to our site. This way, we make sure we're spending our budget wisely and getting the most value for our money.



TRAFFIC SOURCE TRENDING

May 10, 2012
Could you check the trended session volume for "gsearch nonbrand" by week? We want to see if the bid changes made on April 15, 2012, have had any impact on the number of people visiting our site through this channel.

SQL QUERY:

```
select
        min(date(created_at)) as week_start_date,
    count(distinct website_session_id) as sessions
from website_sessions
where created_at < '2012-05-12'
        and utm_source = 'gsearch'
    and utm_campaign = 'nonbrand'
group by yearweek(created_at);
```

Findings:

week_start_date	sessions
2012-03-19	896
2012-03-25	956
2012-04-01	1152
2012-04-08	983
2012-04-15	621
2012-04-22	594
2012-04-29	681
2012-05-06	651

The analysis indicates that "gsearch nonbrand" is quite responsive to bid changes. While aiming for maximum volume, it's essential to strike a balance, ensuring that we don't exceed our ad budget. This finding emphasizes the need for careful bid management to optimize traffic without overspending on advertising.

TRAFFIC SOURCE BID OPTIMIZATION

May 11, 2012
Have you checked the conversion rates from sessions to orders based on different device types, especially comparing mobile and desktop? If desktop performance is better than on mobile, it might be worth considering increasing bids specifically for desktops to attract more traffic.

SQL QUERY:

```
select
        website_sessions.device_type,
    count(distinct website_sessions.website_session_id) as sessions,
    count(distinct orders.website_session_id) as orders,
```

```
    count(distinct orders.website_session_id) / count(distinct website_sessions.website_session_id) as
seassion_to_order_conv_rt
from website_sessions
        left join orders
                        on orders.website_session_id = website_sessions.website_session_id
where website_sessions.created_at < '2012-05-11'
        and utm_source = 'gsearch'
    and utm_campaign = 'nonbrand'
group by device_type;
```

Findings:

device_type	sessions	orders	seassion_to_order_conv_rt
desktop	3911	146	0.0373
mobile	2492	24	0.0096

As a result of the analysis, I've decided to increase our bids on desktop. The rationale is that by bidding higher, we can secure a higher position in the auctions, which, based on our insights, is expected to lead to a boost in sales. This strategic adjustment aims to capitalize on the better performance observed on desktop and enhance our overall sales outcomes.

TRAFFIC SOURCE SEGMENT TRENDING

June 09, 2012
Can you provide weekly trends for both desktop and mobile, starting from April 15, 2012, until the bid change on May 19, 2012? We want to assess the impact of increasing bids on our gsearch nonbrand desktop campaigns and see how it has influenced the volume of website sessions for both desktop and mobile.

SQL QUERY:

```
select
        min(date(created_at)) as week_started_date,
    count(distinct case when device_type = 'mobile' then website_session_id else null end) as mobile_session,
    count(distinct case when device_type = 'desktop' then website_session_id else null end) as desktop_session
from website_sessions
where created_at < '2012-06-09'
        and created_at > '2012-04-15'
        and utm_source = 'gsearch'
    and utm_campaign = 'nonbrand'
group by yearweek(created_at);
```

Findings:

week_started_date	mobile_session	desktop_session
2012-04-15	238	383
2012-04-22	234	360
2012-04-29	256	425
2012-05-06	282	430
2012-05-13	214	403
2012-05-20	190	661
2012-05-27	183	585
2012-06-03	157	582

The analysis reveals that mobile performance has remained relatively stable or slightly declined, while desktop has shown considerable strength. This positive trend in desktop performance is attributed to the bid changes implemented following the previous conversion analysis. The adjustments made to desktop bids have proven effective, contributing to the observed strong performance in terms of website sessions.

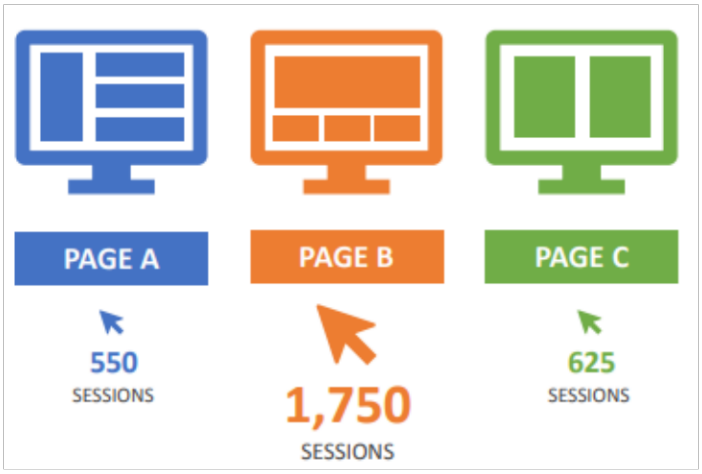
ANALYZING WEBSITE PERFORMANCE:

Website content analysis involves the examination of user engagement to identify key areas for business improvement. This process includes:

1. **Finding Popular Pages:** Determining which pages on your website are viewed most frequently by your users. This helps prioritize content that resonates well with your audience.

2. **Identifying Entry Pages:** Pinpointing the most common entry points to your website, which are the first pages users encounter. Understanding these entry pages is crucial for optimizing the initial user experience.

3. **Assessing Performance:** Evaluating how both the most-viewed pages and common entry pages contribute to your business objectives. This analysis helps align content strategies with overarching business goals, ensuring that the website effectively serves its intended purpose.



IDENTIFYING TOP WEBSITE PAGES

June 09, 2012
Can you provide a list of the most-viewed pages on the website, ranked by the number of sessions? I'd like to better understand which pages are getting the most attention from users.

SQL QUERY:

```
select
        pageview_url,
    count(distinct website_pageview_id) as no_of_pvs
from website_pageviews
where created_at < '2012-06-09'
group by pageview_url
order by no_of_pvs desc;
```

Findings:

pageview_url	no_of_pvs
/home	10403
/products	4239
/the-original-mr-fuzzy	3037
/cart	1306
/shipping	869
/billing	716
/thank-you-for-your-order	306

The analysis indicates that the homepage, products page, and the Mr. Fuzzy page are the primary drivers of our website traffic. These pages collectively attract the bulk of our visitors, signifying their significance in capturing user attention and interest.

IDENTIFYING TOP ENTRY PAGES

June 12, 2012
Could you provide a list of the top entry pages on our website, ranked by the number of users hitting each page first? I want to confirm and understand the most common starting points for our users on the site.

SQL QUERY:

```
create temporary table first_pageviewes
select
        website_session_id,
    min(website_pageview_id) as min_pageview_id
from website_pageviews
where created_at < '2012-06-12'
group by website_session_id;
```

website_session_id	min_pageview_id
1	1
2	2
3	3
4	4
5	5
6	6
7	7
8	12

• • •

```
select
        website_pageviews.pageview_url as landing_page,
    count(first_pageviewes.website_session_id) as sessions_hitting_this_landing_page
from first_pageviewes
        left join website_pageviews
                on website_pageviews.website_pageview_id = first_pageviewes.min_pageview_id
group by landing_page
order by sessions_hitting_this_landing_page
```

Findings:

landing_page	sessions_hitting_this_landing_page
/home	10714

The data reveals that the majority of our traffic enters through the homepage. This clear pattern indicates a key area for improvement. Focusing on enhancements to the homepage would likely have a significant impact on the overall user experience and engagement with our website.

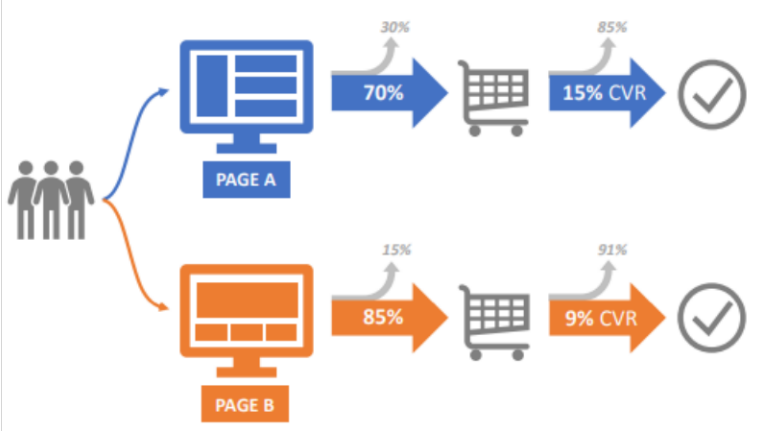
What Is Landing Page Performance & Testing?

Landing page analysis and testing means figuring out how well our important landing pages are doing and trying to make them better. Here's how we do it:

1. **Finding Opportunities:** We look at pages that get lots of visitors but don't keep them interested or make them do what we want. These are the pages we can improve.

2. **Testing Different Versions:** We experiment with these pages while people are using our website. We try different designs or content to see if we can make visitors stay longer or do what we want them to do more often.

3. **Checking the Results:** After the experiments, we see which version of the landing page works better. Then, we decide to use the one that gets us the best results in the future. This helps us make our website more effective and get better results from our visitors.



CALCULATING BOUNCE RATES

June 14, 2012
The other day you showed us that all our traffic is landing on the homepage right now. We should check how that landing page is performing. Can you pull bounce rates for traffic landing on the homepage? I would like to see three numbers...Sessions, Bounced Sessions, and % of Sessions which Bounced (aka "Bounce Rate").

*Refer **Identifying Top Website Pages (Page-10)***

SQL QUERY:

```
-- step 1: Find the first pageview id for relivent seassion

create temporary table first_pageviews
select
        website_pageviews.website_session_id ,
   min(website_pageviews.website_pageview_id) as min_pageview_id
from website_pageviews
        left join website_sessions
                on website_sessions.website_session_id = website_pageviews.website_session_id
where website_pageviews.created_at < '2012-06-14'
group by website_pageviews.website_session_id;
```

website_session_id	min_pageview_id
1	1
2	2
3	3
4	4
5	5
6	6
7	7
8	12
9	14
10	15
11	16
12	17
13	18
14	19
15	20
16	21
17	25
18	26

...

```
-- step 2: Identify the landing page of each session

-- create temporary table sessions_w_landing_page
select
        first_pageviews.website_session_id,
    website_pageviews.pageview_url as landing_page
from website_pageviews
        left join first_pageviews
                on website_pageviews.website_pageview_id = first_pageviews.min_pageview_id
where website_pageviews.pageview_url = '/home';
```

website_session_id	landing_page
1	/home
2	/home
3	/home
4	/home
5	/home
6	/home
7	/home
8	/home
9	/home
10	/home
11	/home
12	/home
13	/home

•••

```
-- step 3: counting pageviewes for each session, to identify "bounces"

create temporary table bounced_sessions
select
        sessions_w_landing_page.landing_page,
        sessions_w_landing_page.website_session_id,
    count(website_pageviews.website_pageview_id) as count_of_page_viewes
from  website_pageviews
        left join sessions_w_landing_page
                on sessions_w_landing_page.website_session_id =
website_pageviews.website_session_id
group by
                sessions_w_landing_page.website_session_id,
                sessions_w_landing_page.landing_page
having count_of_page_viewes = 1;
```

landing_page	website_session_id	count_of_page_viewes
/home	1	1
/home	2	1
/home	3	1
/home	4	1
/home	5	1
/home	7	1
/home	8	1
/home	9	1
/home	10	1
/home	11	1
/home	12	1
/home	13	1
/home	14	1
/home	17	1
/home	19	1
/home	21	1
/home	22	1
/home	23	1

•••

```
-- summarizing by counting total sessions and bounced sessions (final output)

select
        count(distinct sessions_w_landing_page.landing_page) as sesion,
        count(distinct bounced_sessions.website_session_id) as bounced_session,
    count(distinct bounced_sessions.website_session_id) / count(distinct
sessions_w_landing_page.website_session_id) as bounced_rate
from sessions_w_landing_page
        left join bounced_sessions
                on bounced_sessions.website_session_id = sessions_w_landing_page.website_session_id;
```

Findings:

sesion	bounced_session	bounced_rate
1	6538	0.5918

With a nearly 60% bounce rate on the homepage, which is relatively high, especially for paid search, I agree that improvements are needed. To address this, I will create a custom landing page specifically for search and set up an experiment to compare its performance against the current homepage. This proactive approach aims to enhance the user experience and reduce bounce rates, particularly for paid search traffic.

ANALYZING LANDING PAGE TESTS

July 28, 2012

Can you provide the bounce rates for the two groups in our 50/50 test – the new custom landing page (/lander-1) and the homepage (/home) – specifically during the time when /lander-1 was receiving traffic? This will help us evaluate the performance of the new page and make a fair comparison.

SQL QUERY:

```
-- Step 0: Find out when the lander-1 page first created

select
        min(created_at) as first_created_at,
    min(website_pageview_id) as first_pageview_id
from website_pageviews
where pageview_url = '/lander-1'
        and created_at is not null;
```

first_created_at	first_pageview_id
2012-06-19 00:35:54	23504

```
-- step 1: Find the first pageview id for relevant sessions

create temporary table first_test_pageviews
select
```

```
        website_pageviews.website_session_id ,
    min(website_pageviews.website_pageview_id) as min_pageview_id
from website_pageviews
        left join website_sessions
                on website_sessions.website_session_id = website_pageviews.website_session_id
where website_pageviews.created_at < '2012-07-28'
        and website_pageviews.website_pageview_id > 23504
        and website_sessions.utm_source = 'gsearch'
    and website_sessions.utm_campaign = 'nonbrand'
group by website_pageviews.website_session_id;
```

website_session_id	min_pageview_id
11684	23505
11685	23506
11686	23507
11687	23509
11688	23510
11689	23511
11690	23514
11691	23515
11692	23517
11693	23518
11694	23521
11696	23526
11697	23527
11698	23528

...

```
-- step 2: Identify the landing page of home and lander-1 session

create temporary table nonbrand_test_sessions_w_landing_page
select
        first_test_pageviews.website_session_id,
    website_pageviews.pageview_url as landing_page
from first_test_pageviews
        left join website_pageviews
                on website_pageviews.website_pageview_id = first_test_pageviews.min_pageview_id
where website_pageviews.pageview_url in ('/home', '/lander-1');
```

website_session_id	landing_page
11684	/home
11685	/lander-1
11686	/lander-1
11687	/home
11688	/home
11689	/lander-1
11690	/home
11691	/lander-1
11692	/lander-1
11693	/lander-1
11694	/lander-1
11696	/home
11697	/lander-1
11698	/lander-1

...

```
-- step 3: counting pageviews for home and landing-1 session, to identify "bounces"
```

```
create temporary table nonbrand_test_bounced_sessions
select
        nonbrand_test_sessions_w_landing_page.landing_page,
        nonbrand_test_sessions_w_landing_page.website_session_id,
    count(website_pageviews.website_pageview_id) as count_of_page_viewes
from  nonbrand_test_sessions_w_landing_page
        left join website_pageviews
                on nonbrand_test_sessions_w_landing_page.website_session_id =
website_pageviews.website_session_id
group by
                nonbrand_test_sessions_w_landing_page.website_session_id,
                nonbrand_test_sessions_w_landing_page.landing_page
having count_of_page_viewes = 1;
```

landing_page	website_session_id	count_of_page_viewes
/home	11684	1
/lander-1	11685	1
/home	11687	1
/home	11688	1
/home	11690	1
/lander-1	11692	1
/home	11696	1
/lander-1	11697	1
/lander-1	11698	1
/lander-1	11701	1
/lander-1	11702	1
/lander-1	11704	1
/home	11705	1
/home	11706	1

...

```
                -- summarizing by counting total sessions and bounced sessions

select
        nonbrand_test_sessions_w_landing_page.landing_page,
        nonbrand_test_sessions_w_landing_page.website_session_id,
    nonbrand_test_bounced_sessions.website_session_id as bounced_website_session_id
from nonbrand_test_sessions_w_landing_page
        left join nonbrand_test_bounced_sessions
                on nonbrand_test_bounced_sessions.website_session_id =
nonbrand_test_sessions_w_landing_page.website_session_id
order by nonbrand_test_sessions_w_landing_page.website_session_id;
```

landing_page	website_session_id	bounced_website_session_id
/home	11684	11684
/lander-1	11685	11685
/lander-1	11686	NULL
/home	11687	11687
/home	11688	11688
/lander-1	11689	NULL
/home	11690	11690
/lander-1	11691	NULL
/lander-1	11692	11692
/lander-1	11693	NULL
/lander-1	11694	NULL
/home	11696	11696
/lander-1	11697	11697
/lander-1	11698	11698

...

```
-- Final output

select
        nonbrand_test_sessions_w_landing_page.landing_page,
        count(distinct nonbrand_test_sessions_w_landing_page.website_session_id) as sesion,
        count(distinct nonbrand_test_bounced_sessions.website_session_id) as bounced_session,
    count(distinct nonbrand_test_bounced_sessions.website_session_id) / count(distinct
nonbrand_test_sessions_w_landing_page.website_session_id) as bounced_rate
from nonbrand_test_sessions_w_landing_page
        left join nonbrand_test_bounced_sessions
                on nonbrand_test_bounced_sessions.website_session_id =
nonbrand_test_sessions_w_landing_page.website_session_id
group by landing_page;
```

Findings:

first_created_at	first_pageview_id
2012-06-19 00:35:54	23504

landing_page	sesion	bounced_session	bounced_rate
/home	2261	1319	0.5834
/lander-1	2315	1232	0.5322

Fantastic news! The analysis indicates that the custom landing page (/lander-1) has achieved a lower bounce rate compared to the homepage (/home) in our 50/50 test. This outcome marks a success and highlights the positive impact of the new custom landing page on user engagement and retention.

LANDING PAGE TREND ANALYSIS

August 31, 2012
Could you check the weekly trend of paid search nonbrand traffic volumes for the landing pages /home and /lander-1 since June 1st? I need to ensure that the traffic is correctly directed. Additionally, can you provide the weekly trend of overall paid search bounce rates? I'm

interested in understanding if the recent lander change has positively impacted our overall performance.

SQL QUERY:

```
-- Step 1: Find the first pageview id for relevant sessions

CREATE TEMPORARY TABLE session_w_first_pageview_id_and_view_counts AS
SELECT
    website_pageviews.website_session_id,
    MIN(website_pageviews.website_pageview_id) AS first_pageview_id,
    COUNT(website_pageviews.website_pageview_id) AS count_pageview_id
FROM website_pageviews
LEFT JOIN website_sessions
    ON website_sessions.website_session_id = website_pageviews.website_session_id
WHERE website_pageviews.created_at >= '2012-06-01'
    AND website_pageviews.created_at <= '2012-08-31'
    AND website_sessions.utm_source = 'gsearch'
    AND website_sessions.utm_campaign = 'nonbrand'
GROUP BY website_pageviews.website_session_id;
```

website_session_id	first_pageview_id	count_pageview_id
9349	18597	2
9350	18598	3
9351	18600	3
9352	18601	4
9354	18611	1
9356	18616	6
9357	18622	1
9358	18623	3
9359	18626	1
9360	18627	1
9361	18628	1
9363	18631	7
9364	18632	3
9365	18641	1
9366	18642	3

...

```
-- Step 2: Identify the landing page of home and lander-1 sessions

CREATE TEMPORARY TABLE sessions_w_counts_lander_and_created_at
SELECT
    session_w_first_pageview_id_and_view_counts.website_session_id,
    session_w_first_pageview_id_and_view_counts.first_pageview_id,
    session_w_first_pageview_id_and_view_counts.count_pageview_id,
    website_pageviews.pageview_url AS landing_page,
    website_pageviews.created_at AS session_created_at
FROM session_w_first_pageview_id_and_view_counts
LEFT JOIN website_pageviews
    ON website_pageviews.website_pageview_id =
session_w_first_pageview_id_and_view_counts.first_pageview_id;
```

website_session_id	first_pageview_id	count_pageview_id	landing_page	session_created_at
9349	18597	2	/products	2012-06-01 00:02:23
9350	18598	3	/home	2012-06-01 00:05:11
9351	18600	3	/home	2012-06-01 00:06:39
9352	18601	4	/home	2012-06-01 00:08:27
9354	18611	1	/home	2012-06-01 01:08:43
9356	18616	6	/home	2012-06-01 01:37:31
9357	18622	1	/home	2012-06-01 02:29:33
9358	18623	3	/home	2012-06-01 02:39:16
9359	18626	1	/home	2012-06-01 03:12:34
9360	18627	1	/home	2012-06-01 04:09:37
9361	18628	1	/home	2012-06-01 04:10:53
9363	18631	7	/home	2012-06-01 06:11:14
9364	18632	3	/home	2012-06-01 06:12:27
9365	18641	1	/home	2012-06-01 06:33:22
9366	18642	3	/home	2012-06-01 06:39:50

...

-- step 3: counting pageviewes for home and landing-1 session, to identify "bounces"

```
select
        -- yearweek(session_created_at) as year_week,
    min(date(session_created_at)) as week_start_date,
    -- count(distinct website_session_id) as total_sessions,
    -- count(distinct case when count_pageview_id = 1 then website_session_id else null end) as
bounced_session,
    count(distinct case when count_pageview_id = 1 then website_session_id else null end)*1.0 / count(distinct
website_session_id)*1.0 as bounced_rate,
    count(distinct case when landing_page = '/home' then website_session_id else null end) as home_session,
    count(distinct case when landing_page = '/lander-1' then website_session_id else null end) as
landng1_session
from sessions_w_counts_lander_and_created_at
group by
                yearweek(session_created_at);
```

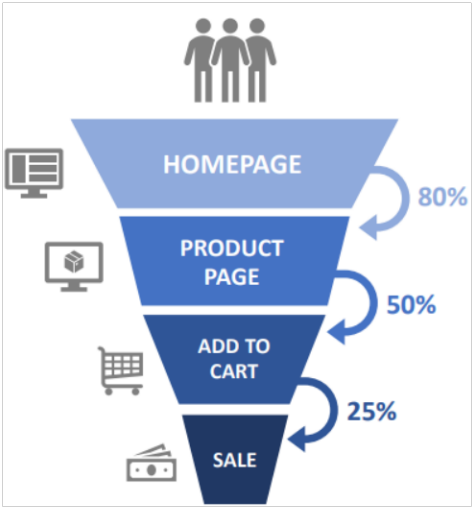
Findings:

week_start_date	bounced_rate	home_session	landng1_session
2012-06-01	0.602273	175	0
2012-06-03	0.587121	792	0
2012-06-10	0.616000	875	0
2012-06-17	0.558195	492	350
2012-06-24	0.582781	369	386
2012-07-01	0.582051	392	388
2012-07-08	0.566792	390	411
2012-07-15	0.542353	429	421
2012-07-22	0.513819	402	394
2012-07-29	0.497082	33	995
2012-08-05	0.538178	0	1087
2012-08-12	0.514028	0	998
2012-08-19	0.500988	0	1012
2012-08-26	0.539016	0	833

The analysis indicates that both /home and /lander-1 received traffic initially, and then we successfully transitioned entirely to the custom lander as planned. Furthermore, there's a positive trend showing a decrease in our overall bounce rate over time.

What is Landing Page Trend Analysis?

Conversion funnel analysis involves thoroughly understanding and enhancing each stage of a user's journey toward buying your products. This process includes identifying the common paths customers follow before making a purchase, tracking the number of users progressing through each step in your conversion flow, and noting where users tend to abandon the process. By optimizing key pain points where users are dropping off, the goal is to increase conversions and boost product sales.



Example:- When we perform conversion funnel analysis, we will look at each step in our conversion flow to see how many customers drop off and how many continue on at each step.

Query: SELECT * FROM website_pageviews WHERE website_session_id = 1059

website_pageview_id	created_at	website_session_id	pageview_url
2039	2012-03-26 13:51:37	1059	/home
2040	2012-03-26 13:54:27	1059	/products
2041	2012-03-26 13:56:48	1059	/the-original-mr-fuzzy
2042	2012-03-26 14:00:14	1059	/cart
2043	2012-03-26 14:04:06	1059	/shipping
2044	2012-03-26 14:05:47	1059	/billing
2045	2012-03-26 14:13:56	1059	/thank-you-for-your-order

BUILDING CONVERSION FUNNELS

September 05, 2012

Can you create a complete conversion funnel for our gsearch visitors, tracking their journey from the new /lander-1 page to placing an order? I would like to understand the drop-off points at

each step. Begin the funnel analysis from /lander-1 and follow through to our thank you page, using data since August 5th.

SQL Quert:

```
-- step 1: select pageview for relevant session
select
        website_sessions.website_session_id,
    website_pageviews.pageview_url,
    website_pageviews.created_at as pageview_created_at,
    case when pageview_url = '/products' then 1 else 0 end as product_page,
    case when pageview_url = '/the-original-mr-fuzzy' then 1 else 0 end as mr_fuzzy_page,
    case when pageview_url = '/cart' then 1 else 0 end as cart_page,
    case when pageview_url = '/shipping' then 1 else 0 end as cshipping_page,
    case when pageview_url = '/billing' then 1 else 0 end as billing_page,
    case when pageview_url = '/thank-you-for-your-order' then 1 else 0 end as thankyou_page
from website_sessions
        left join website_pageviews
                on website_sessions.website_session_id = website_pageviews.website_session_id
where website_sessions.utm_source = 'gsearch'
  and website_sessions.utm_campaign = 'nonbrand'
  and website_sessions.created_at > '2012-08-05'
  and website_sessions.created_at < '2012-09-05'
order by
                website_sessions.website_session_id,
                website_sessions.created_at;
```

website_session_id	pageview_url	pageview_created_at	product_page	mr_fuzzy_page	cart_page	cshipping_page	billing_page	thankyou_page
18243	/lander-1	2012-08-05 00:11:26	0	0	0	0	0	0
18244	/lander-1	2012-08-05 00:44:06	0	0	0	0	0	0
18244	/products	2012-08-05 00:46:01	1	0	0	0	0	0
18244	/the-original-mr-fuzzy	2012-08-05 00:46:32	0	1	0	0	0	0
18244	/cart	2012-08-05 00:47:26	0	0	1	0	0	0
18244	/shipping	2012-08-05 00:51:00	0	0	0	1	0	0
18244	/billing	2012-08-05 00:53:17	0	0	0	0	1	0
18245	/lander-1	2012-08-05 00:47:11	0	0	0	0	0	0
18246	/lander-1	2012-08-05 01:04:50	0	0	0	0	0	0
18246	/products	2012-08-05 01:05:12	1	0	0	0	0	0
18247	/lander-1	2012-08-05 01:15:47	0	0	0	0	0	0
18247	/products	2012-08-05 01:18:25	1	0	0	0	0	0
18247	/the-original-mr-fuzzy	2012-08-05 01:19:03	0	1	0	0	0	0
18249	/lander-1	2012-08-05 02:03:39	0	0	0	0	0	0
18250	/lander-1	2012-08-05 02:38:11	0	0	0	0	0	0

...

```
-- step 2: Create the session level conversion funnel view

-- create temporary table session_lebel_flag_demo
select
        website_session_id,
    max(product_page) as prodect_made_it,
    max(mr_fuzzy_page) as mr_fuzzy_made_it,
    max(cart_page) as cart_made_it,
    max(shipping_page) as shipping_made_it,
    max(billing_page) as billing_made_it,
    max(thankyou_page) as thankyou_made_it
```

```
from (
        select
        website_sessions.website_session_id,
    website_pageviews.pageview_url,
    website_pageviews.created_at as pageview_created_at,
    case when pageview_url = '/products' then 1 else 0 end as product_page,
    case when pageview_url = '/the-original-mr-fuzzy' then 1 else 0 end as mr_fuzzy_page,
    case when pageview_url = '/cart' then 1 else 0 end as cart_page,
    case when pageview_url = '/shipping' then 1 else 0 end as shipping_page,
    case when pageview_url = '/billing' then 1 else 0 end as billing_page,
    case when pageview_url = '/thank-you-for-your-order' then 1 else 0 end as thankyou_page
from website_sessions
        left join website_pageviews
                on website_sessions.website_session_id = website_pageviews.website_session_id
where website_sessions.utm_source = 'gsearch'
  and website_sessions.utm_campaign = 'nonbrand'
  and website_sessions.created_at > '2012-08-05'
  and website_sessions.created_at < '2012-09-05'
order by
                website_sessions.website_session_id,
                website_sessions.created_at ) as pagecview_label
group by website_session_id;
```

website_session_id	prodect_made_it	mr_fuzzy_made_it	cart_made_it	shipping_made_it	billing_made_it	thankyou_made_it
18243	0	0	0	0	0	0
18244	1	1	1	1	1	0
18245	0	0	0	0	0	0
18246	1	0	0	0	0	0
18247	1	1	0	0	0	0
18249	0	0	0	0	0	0
18250	0	0	0	0	0	0
18251	1	1	0	0	0	0
18252	1	1	1	1	1	0
18254	0	0	0	0	0	0
18255	1	1	1	1	1	0
18256	1	1	0	0	0	0
18257	1	1	0	0	0	0
18258	1	0	0	0	0	0
18259	1	1	0	0	0	0

```
                -- step 3: Aggregate the data to assess final performance
select
        count(distinct website_session_id) as sessions,
    count(distinct case when prodect_made_it = 1 then website_session_id else null end) as to_products,
    count(distinct case when mr_fuzzy_made_it = 1 then website_session_id else null end) as to_mr_fuzzy,
    count(distinct case when cart_made_it = 1 then website_session_id else null end) as to_cart,
    count(distinct case when shipping_made_it = 1 then website_session_id else null end) as to_shipping,
    count(distinct case when billing_made_it = 1 then website_session_id else null end) as to_billing,
    count(distinct case when thankyou_made_it = 1 then website_session_id else null end) as to_thankyou
from session_lebel_flag_demo;
```

sessions	to_products	to_mr_fuzzy	to_cart	to_shipping	to_billing	to_thankyou
4493	2115	1567	683	455	361	158

```
-- step 4: Final Output
select
    count(distinct case when prodect_made_it = 1 then website_session_id else null end) /
      count(distinct website_session_id) as lander_click_through_rate,

    count(distinct case when mr_fuzzy_made_it = 1 then website_session_id else null end) /
      count(distinct case when prodect_made_it = 1 then website_session_id else null end) as
product_click_through_rate,

    count(distinct case when cart_made_it = 1 then website_session_id else null end) /
      count(distinct case when mr_fuzzy_made_it = 1 then website_session_id else null end) as
mr_fuzzy_click_through_rate,

    count(distinct case when shipping_made_it = 1 then website_session_id else null end) /
      count(distinct case when cart_made_it = 1 then website_session_id else null end) as
shipping_click_through_rate,

    count(distinct case when billing_made_it = 1 then website_session_id else null end) /
      count(distinct case when shipping_made_it = 1 then website_session_id else null end)as
billing_click_through_rate,

    count(distinct case when thankyou_made_it = 1 then website_session_id else null end) /
      count(distinct case when billing_made_it = 1 then website_session_id else null end) as
thankyou_click_through_rate
from session_lebel_flag_demo;
```

Findings:

lander_click_through_rate	product_click_through_rate	mr_fuzzy_click_through_rate	shipping_click_through_rate	billing_click_through_rate	thankyou_click_through_rate
0.4707	0.7409	0.4359	0.6662	0.7934	0.4377

The analysis indicates that our attention should be directed towards optimizing the /lander-1, Mr. Fuzzy page, and the billing page, as they exhibit the lowest click rates. Specifically, the billing page could benefit from improvements to enhance customer comfort when entering credit card information. I have some ideas for enhancements that I believe will address this concern and potentially improve the conversion rates on the billing page.

ANALYZING CONVERSION FUNNEL TESTS

November 10, 2012

Could you please evaluate whether the updated /billing-2 page, implemented based on our funnel analysis, is performing better than the original /billing page? Specifically, what percentage of sessions on these pages result in placing an order? It's worth noting that this test was conducted across all traffic, not limited to our search visitors.

SQL QUERY:

```
-- step 0: Finding the first time /billing-2 page was seen
```

```
select
        min(created_at) as first_created_at,
        min(website_pageview_id) as first_pageview_id
from website_pageviews
where pageview_url = '/billing-2';
```

first_created_at	first_pageview_id
2012-09-10 00:13:05	53550

```
-- step 1: sellect pageview for relivent session
select
        website_pageviews.website_session_id,
    website_pageviews.pageview_url as billing_virtion_seen,
    orders.order_id
from website_pageviews
        left join orders
                on orders.website_session_id = website_pageviews.website_session_id
where website_pageviews.created_at < '2012-11-10'
        and website_pageviews.website_pageview_id >= 53350
        and website_pageviews.pageview_url in ('/billing','/billing-2')
order by
                website_pageviews.website_session_id,
    website_pageviews.pageview_url;
```

website_session_id	billing_virtion_seen	order_id
25238	/billing	NULL
25241	/billing	868
25268	/billing	NULL
25278	/billing	869
25306	/billing	870
25325	/billing-2	871
25343	/billing	872
25353	/billing	873
25358	/billing-2	874
25368	/billing	875
25393	/billing	876
25411	/billing-2	877
25454	/billing-2	878
25459	/billing-2	NULL
25468	/billing	NULL
		NULL

...

```
-- step 2: Identify each pageview as spesific funnel step
select
        billing_virtion_seen,
    count(distinct website_session_id) as sessions,
    count(distinct order_id) as orders,
    count(distinct order_id) / count(distinct website_session_id) as billing_to_order_rt
from (
```

```
        select
        website_pageviews.website_session_id,
    website_pageviews.pageview_url as billing_virtion_seen,
    orders.order_id
from website_pageviews
        left join orders
                on orders.website_session_id = website_pageviews.website_session_id
where website_pageviews.created_at < '2012-11-10'
        and website_pageviews.website_pageview_id >= 53350
        and website_pageviews.pageview_url in ('/billing','/billing-2')
order by
                website_pageviews.website_session_id,
    website_pageviews.pageview_url
        ) as billing_session_w_order
group by billing_virtion_seen;
```

Findings:

billing_virtion_seen	sessions	orders	billing_to_order_rt
/billing	662	303	0.4577
/billing-2	654	410	0.6269

The updated billing page, /billing-2, is showing significant improvement in converting customers. This positive outcome prompts the decision to promptly implement the new version for all our customers. I will coordinate with Engineering to roll out this enhancement without delay.

ANALYSIS FOR CHANNEL MANAGEMENT

What Is Channel Portfolio Optimization?

Analyzing a portfolio of marketing channels involves strategically managing your marketing budget by bidding efficiently and leveraging data to enhance effectiveness. This process includes identifying which marketing channels drive the most sessions and orders to your website. Furthermore, it entails understanding variations in user characteristics and conversion rates across different marketing channels. By optimizing bids and allocating marketing spend across a diverse portfolio of channels, the objective is to achieve maximum performance and return on investment.

Common use cases:

• Understanding which marketing channels are driving the most sessions and orders through your website

• Understanding differences in user characteristics and conversion performance across marketing channels

• Optimizing bids and allocating marketing spend across a multi-channel portfolio to achieve maximum performance

TRACKING PARAMETERS USED:

When businesses run paid marketing campaigns, they often obsess over performance and measure everything; how much they spend, how well traffic converts to sales, etc.
Paid traffic is commonly tagged with tracking (UTM) parameters, which are appended to URLs and allow us to tie website activity back to specific traffic sources and campaigns.

Example:

SELECT DISTINCT utm_source, utm_campaign FROM website_sessions

www.abcwebsite.com?utm_source=trafficSource&utm_campaign=campaignName

utm_source	utm_campaign
NULL	NULL
bsearch	brand
bsearch	nonbrand
gsearch	brand
gsearch	nonbrand
socialbook	desktop_targeted
socialbook	pilot

ANALYZING CHANNEL PORTFOLIOS

November 29, 2012
Could you provide the weekly trend of session volumes for the newly launched bsearch paid search channel since its inception around August 22nd? Additionally, can you compare this data to the weekly trend of gsearch nonbrand sessions? I'm keen on understanding the significance of bsearch for our business in relation to the well-performing gsearch.

SQL QUERY:

```
SELECT
    min(date(created_at)) as week_start_date,
    count(distinct website_sessions.website_session_id) as sessions,
    count(distinct case when utm_source = 'gsearch' then website_session_id else null end) as gsearch_session,
    count(distinct case when utm_source = 'bsearch' then website_session_id else null end) as bsearch_session
FROM website_sessions
WHERE created_at BETWEEN '2012-08-22' AND '2012-11-29'
          and utm_campaign = 'nonbrand'
GROUP BY yearweek(created_at);
```

Findings:

week_start_date	sessions	gsearch_session	bsearch_session
2012-08-22	787	590	197
2012-08-26	1399	1056	343
2012-09-02	1215	925	290
2012-09-09	1280	951	329
2012-09-16	1516	1151	365
2012-09-23	1371	1050	321
2012-09-30	1315	999	316
2012-10-07	1332	1002	330
2012-10-14	1677	1257	420
2012-10-21	1733	1302	431
2012-10-28	1595	1211	384
2012-11-04	1779	1350	429
2012-11-11	1684	1246	438
2012-11-18	4601	3508	1093
2012-11-25	3060	2286	774

The analysis reveals that bsearch tends to attract approximately one-third of the traffic compared to gsearch. This notable difference indicates the importance of delving deeper into understanding the bsearch channel, as it has a significant potential impact on our business.

COMPARING CHANNEL CHARACTERISTICS

November 30, 2012
Can you provide insights into the bsearch nonbrand campaign by pulling the percentage of traffic coming from Mobile since its inception around August 22nd? Additionally, please

compare this Mobile traffic percentage with that of gsearch. Feel free to explore and share any other noteworthy findings. Aggregate data since August 22nd is sufficient for this analysis; there's no need to present trending information at this point.

SQL QUERY:

```
SELECT
   utm_source,
   count(distinct website_sessions.website_session_id) as sessions,
   count(distinct case when device_type = 'mobile' then website_session_id else null end) as mobile_session,
   count(distinct case when device_type = 'mobile' then website_session_id else null end) /
     count(distinct website_sessions.website_session_id) as percent_of_mobile_sessions
FROM website_sessions
WHERE created_at BETWEEN '2012-08-22' AND '2012-11-30'
   and utm_campaign = 'nonbrand'
GROUP BY utm_source;
```

Findings:

utm_source	sessions	mobile_session	percent_of_mobile_sessions
bsearch	6522	562	0.0862
gsearch	20073	4921	0.2452

The analysis of desktop to mobile splits reveals intriguing differences between the bsearch and gsearch channels, especially in terms of device usage. This insight emphasizes the distinct nature of these channels from a device standpoint. Moving forward, it's essential to keep this understanding in mind as we continue to learn and optimize, acknowledging that these channels present notable differences.

CROSS CHANNEL BID OPTIMIZATION

December 01, 2012
Can you retrieve and compare the nonbrand conversion rates from session to order for gsearch and bsearch, while also segmenting the data by device type? I'm particularly interested in the analysis covering the period from August 22 to September 18. It's important to note that we initiated a special pre-holiday campaign for gsearch starting on September 19th, so the data after that date is not applicable for this comparison.

SQL QUERY:

```
SELECT
        device_type,
   utm_source,
   count(distinct website_sessions.website_session_id) as sessions,
   count(distinct orders.order_id) as orders,
```

```
    count(distinct orders.order_id) / count(distinct website_sessions.website_session_id) as
orders_convertion_rate
FROM website_sessions
        left join orders
                on orders.website_session_id = website_sessions.website_session_id
WHERE website_sessions.created_at between '2012-08-22' AND '2012-09-19'
    and utm_campaign = 'nonbrand'
GROUP BY device_type, utm_source;
```

Findings:

device_type	utm_source	sessions	orders	orders_convertion_rate
desktop	bsearch	1162	44	0.0379
desktop	gsearch	3011	136	0.0452
mobile	bsearch	130	1	0.0077
mobile	gsearch	1015	13	0.0128

Confirming my suspicion, the analysis indicates that the gsearch and bsearch channels do not perform identically. To optimize our overall paid marketing budget, it's prudent to differentiate our bids. As a result, I will adjust the bids for bsearch downward, considering its under-performance in comparison to gsearch.

CHANNEL PORTFOLIO TRENDS

December 22, 2012

Could you retrieve the weekly session volume for gsearch and bsearch nonbrand, segmented by device, since November 4th? Additionally, it would be helpful to include a comparison metric that illustrates bsearch as a percentage of gsearch for each device. This information will aid in assessing the impact of bidding down bsearch nonbrand on December 2nd.

SQL QUERY:

```
SELECT
        min(date(created_at)) as min_start_date,
    count(distinct case when utm_source = 'gsearch' and device_type = 'desktop' then website_session_id else
null end) as g_desktop_session_id,
    count(distinct case when utm_source = 'bsearch' and device_type = 'desktop' then website_session_id else
null end) as b_desktop_session_id,
    count(distinct case when utm_source = 'bsearch' and device_type = 'desktop' then website_session_id else
null end) /
                        count(distinct case when utm_source = 'gsearch' and device_type = 'desktop' then
website_session_id else null end) as b_pct_of_g_dtop,

        count(distinct case when utm_source = 'gsearch' and device_type = 'mobile' then website_session_id
else null end) as g_mobile_session_id,
```

```
    count(distinct case when utm_source = 'bsearch' and device_type = 'mobile' then website_session_id else
null end) as b_mobile_session_id,
    count(distinct case when utm_source = 'bsearch' and device_type = 'mobile' then website_session_id else
null end) /
                                count(distinct case when utm_source = 'gsearch' and device_type = 'mobile' then
website_session_id else null end) as b_pct_of_g_mob
FROM website_sessions
WHERE website_sessions.created_at between '2012-11-04' AND '2012-12-22'
    and utm_campaign = 'nonbrand'
GROUP BY yearweek(created_at);
```

Findings:

min_start_date	g_desktop_session_id	b_desktop_session_id	b_pct_of_g_dtop	g_mobile_session_id	b_mobile_session_id	b_pct_of_g_mob
2012-11-04	1027	400	0.3895	323	29	0.0898
2012-11-11	956	401	0.4195	290	37	0.1276
2012-11-18	2655	1008	0.3797	853	85	0.0996
2012-11-25	2058	843	0.4096	692	62	0.0896
2012-12-02	1326	517	0.3899	396	31	0.0783
2012-12-09	1277	293	0.2294	424	46	0.1085
2012-12-16	1270	348	0.2740	376	41	0.1090

The data shows a decline in bsearch traffic following the bid down, and interestingly, gsearch also experienced a decrease after Black Friday and Cyber Monday, though to a lesser extent. This seems acceptable, especially considering the lower conversion rate of bsearch.

What is analyzing direct traffic?

Analyzing branded or direct traffic is essential for gauging the success and influence of your brand on consumers and business performance. This involves assessing the revenue generated from direct traffic, which often translates to high-margin revenue without incurring customer acquisition costs. Additionally, it aims to determine whether paid traffic contributes to a "halo" effect, encouraging additional direct traffic. This analysis is crucial for evaluating the impact of various initiatives on how many customers actively seek out your business.

COMMON USE CASES:

• Identifying how much revenue you are generating from direct traffic – this is high margin revenue without a direct cost of customer acquisition

• Understanding whether your paid traffic is generating a “halo” effect, and promoting additional direct traffic

• Assessing the impact of various initiatives on how many customers seek out your business

ANALYZING FREE CHANNELS

December 23, 2012

Can you provide a breakdown of organic search, direct type-in, and paid brand search sessions by month? Additionally, could you express these sessions as a percentage of paid search nonbrand sessions? This information will help address a potential investor's inquiry regarding whether we are building momentum with our brand or if we continue to heavily rely on paid traffic.

SQL QUERY:

```
select
        year(created_at) as yr,
   month(created_at) as mon,
   count(distinct case when type_of_trafic = 'paid_nonbrand' then website_session_id else null end) as
nonbrand,
   count(distinct case when type_of_trafic = 'paid_brand' then website_session_id else null end) as brand,
   count(distinct case when type_of_trafic = 'paid_brand' then website_session_id else null end) /
                        count(distinct case when type_of_trafic = 'paid_nonbrand' then website_session_id
else null end) as brand_percent_of_nonbrand,
        count(distinct case when type_of_trafic = 'direct_type_in' then website_session_id else null end) as
direct,
   count(distinct case when type_of_trafic = 'direct_type_in' then website_session_id else null end) /
                        count(distinct case when type_of_trafic = 'paid_nonbrand' then website_session_id
else null end) as direct_percent_of_nonbrand,
        count(distinct case when type_of_trafic = 'organic_search_trafic' then website_session_id else null
end) as organic,
   count(distinct case when type_of_trafic = 'organic_search_trafic' then website_session_id else null end) /
                        count(distinct case when type_of_trafic = 'paid_nonbrand' then website_session_id
else null end) as organic_percent_of_nonbrand
from(
select distinct
        website_session_id,
   created_at,
   case
                when utm_source is null and http_referer in ('https://www.gsearch.com',
'https://www.bsearch.com')  then 'organic_search_trafic'
      when utm_campaign = 'nonbrand' then 'paid_nonbrand'
      when utm_campaign = 'brand' then 'paid_brand'
      when utm_source is null      and http_referer is null then 'direct_type_in'
```

```
                else 'other'
    end as type_of_trafic
from website_sessions
where created_at < '2012-12-23'
) as session_w_channel_group
group by 1,2
```

Findings:

yr	mon	nonbrand	brand	brand_percent_of_nonbrand	direct	direct_percent_of_nonbrand	organic	organic_percent_of_nonbrand
2012	3	1852	10	0.0054	9	0.0049	8	0.0043
2012	4	3509	76	0.0217	71	0.0202	78	0.0222
2012	5	3295	140	0.0425	151	0.0458	150	0.0455
2012	6	3439	164	0.0477	170	0.0494	190	0.0552
2012	7	3660	195	0.0533	187	0.0511	207	0.0566
2012	8	5318	264	0.0496	250	0.0470	265	0.0498
2012	9	5591	339	0.0606	285	0.0510	331	0.0592
2012	10	6883	432	0.0628	440	0.0639	428	0.0622
2012	11	12260	556	0.0454	571	0.0466	624	0.0509
2012	12	6643	464	0.0698	482	0.0726	492	0.0741

The analysis reveals positive trends – our brand, direct, and organic traffic volumes are not only increasing, but they are also growing as a percentage of our overall paid traffic volume. This suggests a promising development, indicating that our brand is gaining traction organically and becoming a more significant driver of traffic.

BUSINESS PATTERNS & SEASONALITY:

Analyzing business patterns involves extracting valuable insights to enhance efficiency and proactively anticipate future trends. This process encompasses day-parting analysis, which helps in determining the optimal level of support staff required at different times of the day or days of the week. Additionally, it includes the examination of seasonality patterns to prepare for anticipated spikes or slowdowns in demand, allowing for strategic planning and resource allocation.

COMMON USE CASES:

• Day-parting analysis to understand how much support staff you should have at different times of day or days of the week

• Analyzing seasonality to better prepare for upcoming spikes or slowdowns in demand

ANALYZING SEASONALITY

January 02, 2013

Can you retrieve the monthly and weekly session volume and order volume data from 2012 for our analysis? As we experienced success in that year, examining the patterns may help us identify any seasonal trends that could be valuable for planning in 2013. Your assistance in pulling this data would be greatly appreciated.

SQL QUERY:

```
select
        year(website_sessions.created_at) as years,
    month(website_sessions.created_at) as months,
    week(website_sessions.created_at) as weeks,
    day(website_sessions.created_at) as dates,
    min(date(website_sessions.created_at)) as start_of_week,
    count(distinct website_sessions.website_session_id) as sessions,
    count(distinct orders.order_id) as orders
from website_sessions
        left join orders
                on orders.website_session_id = website_sessions.website_session_id
where website_sessions.created_at < '2013-01-01'
group by 1,2,3,4;
```

Findings:

week_start_date	sessions	orders
2012-03-19	896	25
2012-03-25	983	35
2012-04-01	1193	29
2012-04-08	1029	28
2012-04-15	679	22
2012-04-22	655	18
2012-04-29	770	19
2012-05-06	798	17
2012-05-13	706	23
2012-05-20	965	28
2012-05-27	875	31
2012-06-03	920	34
2012-06-10	994	29
2012-06-17	966	37
2012-06-24	883	32
2012-07-01	892	30
2012-07-08	925	36
2012-07-15	987	47
2012-07-22	954	41
2012-07-29	1172	55
2012-08-05	1235	48
2012-08-12	1181	39
2012-08-19	1522	55
2012-08-26	1593	52
2012-09-02	1418	56
2012-09-09	1488	72
2012-09-16	1776	76
2012-09-23	1624	70
2012-09-30	1553	67
2012-10-07	1632	73
2012-10-14	1955	93
2012-10-21	2042	95
2012-10-28	1923	82
2012-11-04	2086	91
2012-11-11	1973	101
2012-11-18	5130	223
2012-11-25	4172	179
2012-12-02	2727	145
2012-12-09	2489	123
2012-12-16	2718	135
2012-12-23	1682	74
2012-12-30	309	21

•••

The analysis indicates a consistent growth throughout the year, with notable spikes in volume observed during the holiday months, particularly in the weeks surrounding Black Friday and Cyber Monday. This steady growth pattern, coupled with the holiday peaks, provides valuable insights for understanding our business trends in 2012.

ANALYZING BUSINESS PATTERNS

January 05, 2013

Can you analyze the average website session volume, considering both the hour of the day and the day of the week, within the date range of September 15 to November 15, 2013? This information will be instrumental in helping us determine the appropriate staffing levels for potential live chat support, enhancing our customer experience. To ensure accuracy, let's exclude the holiday time period from this analysis.

SQL QUERY:

```
SELECT
   hr,
   ROUND(AVG(CASE WHEN week_of_day = 0 THEN sessions ELSE NULL END), 1) AS Monday,
   ROUND(AVG(CASE WHEN week_of_day = 1 THEN sessions ELSE NULL END), 1) AS Tuesday,
   ROUND(AVG(CASE WHEN week_of_day = 2 THEN sessions ELSE NULL END), 1) AS Wednesday,
   ROUND(AVG(CASE WHEN week_of_day = 3 THEN sessions ELSE NULL END), 1) AS Thursday,
   ROUND(AVG(CASE WHEN week_of_day = 4 THEN sessions ELSE NULL END), 1) AS Friday,
   ROUND(AVG(CASE WHEN week_of_day = 5 THEN sessions ELSE NULL END), 1) AS Saturday,
   ROUND(AVG(CASE WHEN week_of_day = 6 THEN sessions ELSE NULL END), 1) AS Sunday
FROM (
   SELECT
      DATE(created_at) AS created_date,
      WEEKDAY(created_at) AS week_of_day,
      HOUR(created_at) AS hr,
      COUNT(DISTINCT website_session_id) AS sessions
   FROM website_sessions
   WHERE created_at BETWEEN '2012-09-15' AND '2012-11-15'
   GROUP BY 1, 2, 3
) AS daily_hourly_session
GROUP BY hr;
```

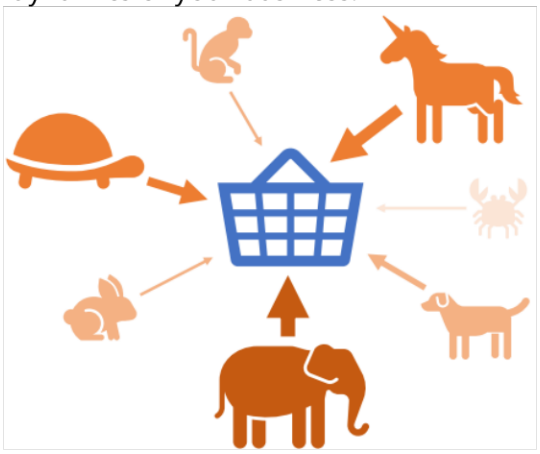
Findings:

hr	mon	tue	wed	thu	fri	sat	sun
0	8.7	7.7	6.3	7.4	6.8	5.0	5.0
1	6.6	6.7	5.3	4.9	7.1	5.0	3.0
2	6.1	4.4	4.4	6.1	4.6	3.7	3.0
3	5.7	4.0	4.7	4.6	3.6	3.9	3.4
4	5.9	6.3	6.0	4.0	6.1	2.8	2.4
5	5.0	5.4	5.1	5.4	4.6	4.3	3.9
6	5.4	5.6	4.8	6.0	6.8	4.0	2.6
7	7.3	7.8	7.4	10.6	7.0	5.7	4.8
8	12.3	12.2	13.0	16.5	10.5	4.3	4.1
9	17.6	15.7	19.6	19.3	17.5	7.6	6.0
10	18.4	17.7	21.0	18.4	19.0	8.3	6.3
11	18.0	19.1	24.9	21.6	20.9	7.2	7.7
12	21.1	23.3	22.8	24.1	19.0	8.6	6.1
13	17.8	23.0	20.8	20.6	21.6	8.1	8.4
14	17.9	21.6	22.3	18.5	19.5	8.7	6.7
15	21.6	17.1	25.3	23.5	21.3	6.9	7.1
16	21.1	23.7	23.7	19.6	20.9	7.6	6.6
17	19.4	15.9	20.2	19.8	12.9	6.4	7.6
18	12.7	15.0	14.8	15.3	10.9	5.3	6.8
19	12.4	14.1	13.3	11.6	14.3	7.1	6.4
20	12.1	12.4	14.2	10.6	10.3	5.7	8.4
21	9.1	12.6	11.4	9.4	7.3	5.7	10.2
22	9.1	10.0	9.8	12.1	6.0	5.7	10.2
23	8.8	8.6	9.6	10.6	7.6	5.3	8.3

Based on discussions with support companies, it appears that staffing one support staff member around the clock should align with an average of ~10 sessions per hour per employee. Additionally, to ensure optimal coverage during peak hours, we can plan to double up with two staff members from 8 am to 5 pm, Monday through Friday. This approach aims to provide efficient support and enhance the overall customer experience.

PRODUCT ANALYSIS:

Analyzing product sales is crucial for gaining insights into each product's contribution to your business and assessing the impact of product launches on the overall portfolio. This involves scrutinizing sales and revenue data specific to each product, monitoring the influence of adding a new product to the portfolio, and observing sales trends to gauge the overall health of your business. Essentially, it provides a comprehensive understanding of how individual products shape the performance and dynamics of your business.



COMMON USE CASES:

- Analyzing sales and revenue by product
- Monitoring the impact of adding a new product to your product portfolio
- Watching product sales trends to understand the overall health of your business

PRODUCT LEVEL SALES ANALYSIS

January 04, 2013

Can you retrieve and present the monthly trends to date for our current flagship product, specifically focusing on the number of sales, total revenue, and total margin generated for the business? This deep dive will provide valuable insights as we prepare to launch a new product.

SQL QUERY:

```
select
        year(created_at) as yr,
    month(created_at) as mo,
    count(distinct order_id) as no_of_sales,
    sum(price_usd) as total_ravinue,
    sum(price_usd - cogs_usd) as total_margin
from orders
where created_at < '2013-01-04'
group by
        yr,
    mo;
```

Findings:

yr	mo	no_of_sales	total_ravinue	total_margin
2012	3	60	2999.40	1830.00
2012	4	99	4949.01	3019.50
2012	5	108	5398.92	3294.00
2012	6	140	6998.60	4270.00
2012	7	169	8448.31	5154.50
2012	8	228	11397.72	6954.00
2012	9	287	14347.13	8753.50
2012	10	371	18546.29	11315.50
2012	11	618	30893.82	18849.00
2012	12	506	25294.94	15433.00
2013	1	42	2099.58	1281.00

The gathered data serves as an excellent baseline, offering insights into how our revenue and margin evolve with the rollout of the new product. Additionally, it provides a clear view of our growth pattern in general, allowing us to track and understand the performance dynamics of our flagship product over time.

PRODUCT LAUNCH SALES ANALYSIS

April 05, 2013

Could you please compile trended analysis data since April 1, 2013, encompassing monthly order volume, overall conversion rates, revenue per session, and a detailed breakdown of sales

by product? This information will offer valuable insights into the performance and impact of our second product since its launch on January 6th.

SQL QUERY:

```
select
        year(website_sessions.created_at) as yr,
    month(website_sessions.created_at) as mo,
    count(distinct website_sessions.website_session_id) as sessions,
    count(distinct orders.order_id) as orders,
    count(distinct orders.order_id) / count(distinct website_sessions.website_session_id) as convertion_rate,
    sum(price_usd) / count(distinct website_sessions.website_session_id) as revinue_per_session,
    count(distinct case when primary_product_id = 1 then order_id else null end) as product_one_order,
    count(distinct case when primary_product_id = 2 then order_id else null end) as product_two_order
from website_sessions
        left join orders
                on website_sessions.website_session_id = orders.website_session_id
where website_sessions.created_at < '2013-04-01'
        and website_sessions.created_at > '2012-04-01'
group by
        yr,
    mo;
```

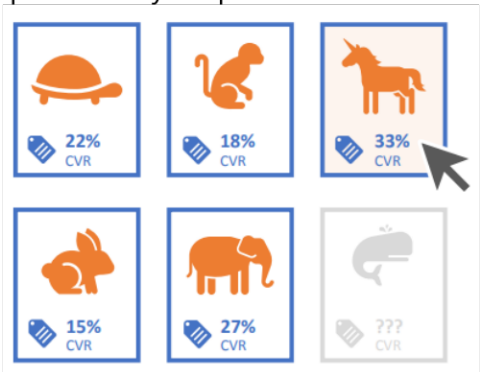
Findings:

yr	mo	sessions	orders	convertion_rate	revinue_per_session	product_one_order	product_two_order
2012	4	3734	99	0.0265	1.325391	99	0
2012	5	3736	108	0.0289	1.445107	108	0
2012	6	3963	140	0.0353	1.765985	140	0
2012	7	4249	169	0.0398	1.988305	169	0
2012	8	6097	228	0.0374	1.869398	228	0
2012	9	6546	287	0.0438	2.191740	287	0
2012	10	8183	371	0.0453	2.266441	371	0
2012	11	14011	618	0.0441	2.204969	618	0
2012	12	10072	506	0.0502	2.511412	506	0
2013	1	6401	391	0.0611	3.127025	344	47
2013	2	7168	497	0.0693	3.692108	335	162
2013	3	6264	385	0.0615	3.176269	320	65

The analysis confirms a positive trend, showcasing improvements in conversion rates and revenue per session over time, which is indeed promising. However, it poses a challenge in discerning whether the growth since January is primarily attributed to the new product launch or reflects a continuation of overall business improvements. Further investigation may be necessary to isolate the specific factors contributing to this positive trajectory.

What is PRODUCT LEVEL WEBSITE ANALYSIS?

Product-focused website analysis involves gaining insights into how customers interact with individual products and assessing the conversion effectiveness of each product. This includes identifying which products generate the most interest on multi-product showcase pages, analyzing the impact on website conversion rates when introducing a new product, and constructing product-specific conversion funnels to discern whether certain products exhibit better conversion rates than others. Essentially, it helps in understanding customer behavior and optimizing strategies for each product in your portfolio.



COMMON USE CASES:

- Understanding which of your products generate the most interest on multi-product showcase pages
- Analyzing the impact on website conversion rates when you add a new product
- Building product-specific conversion funnels to understand whether certain products convert better than others.

PRODUCT PATHING ANALYSIS

April 06, 2014

Can you retrieve the clickthrough rates from the /products page since the launch of the new product on January 6th, 2013, broken down by each product? Additionally, I'd like to compare these rates to the three months leading up to the launch to establish a baseline. This analysis will help us understand the user path and conversion funnel dynamics following the introduction of the new product.

SQL QUERY:

```
                                    -- step-1: find the relivent /product pageview
create temporary table products_pageview
select
        website_session_id,
    website_pageview_id,
    created_at,
    case
                when created_at < '2013-01-06' then 'A.Pre_Product_2'
        when created_at >= '2013-01-06' then 'B.Post_Product_2'
        else 'check_logic'
```

```
        end as time_period
from website_pageviews
where created_at < '2013-04-06'
        and created_at >= '2012-10-06' -- start of 3 month before product 2 launch
    and pageview_url = '/products';
```

website_session_id	website_pageview_id	created_at	time_period
31517	67216	2012-10-06 00:01:26	A.Pre_Product_2
31518	67220	2012-10-06 00:20:27	A.Pre_Product_2
31519	67222	2012-10-06 00:24:31	A.Pre_Product_2
31521	67227	2012-10-06 00:48:54	A.Pre_Product_2
31524	67232	2012-10-06 01:50:14	A.Pre_Product_2
31525	67236	2012-10-06 02:11:18	A.Pre_Product_2
31528	67240	2012-10-06 03:27:07	A.Pre_Product_2
31532	67250	2012-10-06 05:04:15	A.Pre_Product_2
31534	67254	2012-10-06 05:10:50	A.Pre_Product_2
31536	67261	2012-10-06 05:40:33	A.Pre_Product_2
31545	67272	2012-10-06 07:58:34	A.Pre_Product_2
31549	67278	2012-10-06 08:53:11	A.Pre_Product_2
31551	67283	2012-10-06 09:00:13	A.Pre_Product_2
31552	67284	2012-10-06 09:01:01	A.Pre_Product_2
31559	67292	2012-10-06 10:02:06	A.Pre_Product_2

...

```
        -- step-2: finding the next pageview id that occurs AFTER the product pageview
create temporary table session_w_next_pageview_id
select
        products_pageview.time_period,
    products_pageview.website_session_id,
    min(website_pageviews.website_pageview_id) as min_next_pageview_id
from products_pageview
        left join website_pageviews
                on website_pageviews.website_session_id = products_pageview.website_session_id
        and website_pageviews.website_pageview_id > products_pageview.website_pageview_id
group by 1,2;
```

time_period	website_session_id	min_next_pageview_id
A.Pre_Product_2	31517	67217
A.Pre_Product_2	31518	67221
A.Pre_Product_2	31519	67223
A.Pre_Product_2	31521	67228
A.Pre_Product_2	31524	67233
A.Pre_Product_2	31525	67237
A.Pre_Product_2	31528	67241
A.Pre_Product_2	31532	67253
A.Pre_Product_2	31534	67255
A.Pre_Product_2	31536	67262
A.Pre_Product_2	31545	67273
A.Pre_Product_2	31549	67281
A.Pre_Product_2	31551	NULL
A.Pre_Product_2	31552	NULL
A.Pre_Product_2	31559	67293

...

```
        -- step-3: find the pageview_url associeted with any applicable next pageview id
create temporary table session_w_next_pageview_url
select
        session_w_next_pageview_id.time_period,
    session_w_next_pageview_id.website_session_id,
    website_pageviews.pageview_url as next_pageview_url
from session_w_next_pageview_id
        left join website_pageviews
```

```
        on website_pageviews.website_pageview_id =
session_w_next_pageview_id.min_next_pageview_id;
```

time_period	website_session_id	next_pageview_url
A.Pre_Product_2	31517	/the-original-mr-fuzzy
A.Pre_Product_2	31518	/the-original-mr-fuzzy
A.Pre_Product_2	31519	/the-original-mr-fuzzy
A.Pre_Product_2	31521	/the-original-mr-fuzzy
A.Pre_Product_2	31524	/the-original-mr-fuzzy
A.Pre_Product_2	31525	/the-original-mr-fuzzy
A.Pre_Product_2	31528	/the-original-mr-fuzzy
A.Pre_Product_2	31532	/the-original-mr-fuzzy
A.Pre_Product_2	31534	/the-original-mr-fuzzy
A.Pre_Product_2	31536	/the-original-mr-fuzzy
A.Pre_Product_2	31545	/the-original-mr-fuzzy
A.Pre_Product_2	31549	/the-original-mr-fuzzy
A.Pre_Product_2	31551	NULL
A.Pre_Product_2	31552	NULL
A.Pre_Product_2	31559	/the-original-mr-fuzzy

•••

```
        -- step-4: summarizing the data and analizing Pre vs. Post Pageviews
select
        time_period,
   count(distinct website_session_id) as sessions,
   count(distinct case when next_pageview_url is not null then website_session_id else not null end) as
w_next_pg,
   count(distinct case when next_pageview_url is not null then website_session_id else not null end) /
     count(distinct website_session_id) as pct_w_next_pg,
   count(distinct case when next_pageview_url = '/the-original-mr-fuzzy' then website_session_id else not
null end) as to_mrfuzzy,
   count(distinct case when next_pageview_url = '/the-original-mr-fuzzy' then website_session_id else not
null end) /
     count(distinct website_session_id)as pct_to_mrfuzzy,
   count(distinct case when next_pageview_url = '/the-forever-love-bear' then website_session_id else not
null end) as to_lovebear,
   count(distinct case when next_pageview_url = '/the-forever-love-bear' then website_session_id else not
null end) /
     count(distinct website_session_id) as pct_to_lovebear
from session_w_next_pageview_url
group by time_period;
```

Findings:

time_period	sessions	w_next_pg	pct_w_next_pg	to_mrfuzzy	pct_to_mrfuzzy	to_lovebear	pct_to_lovebear
A.Pre_Product_2	15696	11347	0.7229	11347	0.7229	0	0.0000
B.Post_Product_2	10709	8200	0.7657	6654	0.6213	1546	0.1444

The analysis indicates that the percentage of /products pageviews clicking through to Mr. Fuzzy has decreased since the launch of the Love Bear. However, the overall clickthrough rate has increased, suggesting that the new product, Love Bear, is generating additional interest in our products. This information provides valuable insights into the shifting dynamics of user engagement and interest following the introduction of the new product.

PRODUCT CONVERISON FUNNELS

April 10, 2014

Could you please analyze and compare the conversion funnels for our two products, Love Bear and Mr. Fuzzy, since January 6th? I'm particularly interested in understanding the journey from each product page to the final conversion for all website traffic. This comparison will provide insights into the performance and effectiveness of the conversion paths for each product.

SQL QUERY:

```
-- step-1: select all the pageview for relivent session
create temporary table session_seeing_products_pages
select
        website_session_id,
    website_pageview_id,
    pageview_url as product_page_seen
from website_pageviews
where created_at < '2013-04-10'
        and created_at >= '2013-1-06' -- product 2 lonch date
    and pageview_url in ('/the-original-mr-fuzzy' , '/the-forever-love-bear');
```

website_session_id	website_pageview_id	product_page_seen
63513	138944	/the-original-mr-fuzzy
63515	138952	/the-original-mr-fuzzy
63516	138956	/the-original-mr-fuzzy
63517	138959	/the-original-mr-fuzzy
63518	138962	/the-original-mr-fuzzy
63519	138966	/the-original-mr-fuzzy
63520	138969	/the-original-mr-fuzzy
63521	138972	/the-original-mr-fuzzy
63526	138986	/the-original-mr-fuzzy
63527	138989	/the-original-mr-fuzzy
63528	138992	/the-original-mr-fuzzy
63529	138995	/the-original-mr-fuzzy
63532	139002	/the-original-mr-fuzzy
63533	139004	/the-original-mr-fuzzy
63534	139008	/the-original-mr-fuzzy

...

```
-- step-2: finding out which pageview urls look for
select
        distinct website_pageviews.pageview_url
from session_seeing_products_pages
        left join website_pageviews
                on website_pageviews.website_session_id =
session_seeing_products_pages.website_session_id
      and website_pageviews.website_pageview_id > session_seeing_products_pages.website_pageview_id;
```

pageview_url
/cart
/shipping
/billing-2
/thank-you-for-your-order
NULL

```
-- step-3: pull all pageviews and identify the funnel step
select
```

```
        session_seeing_products_pages.website_session_id,
    session_seeing_products_pages.product_page_seen,
    case when pageview_url = '/cart' then 1 else 0 end as cart_page,
    case when pageview_url = '/shipping' then 1 else 0 end as shipping_page,
    case when pageview_url = '/billing-2' then 1 else 0 end as billing2_page,
    case when pageview_url = '/thank-you-for-your-order' then 1 else 0 end as thankyou_page
from session_seeing_products_pages
    left join website_pageviews
            on website_pageviews.website_session_id =
session_seeing_products_pages.website_session_id
        and website_pageviews.website_pageview_id > session_seeing_products_pages.website_pageview_id
order by 1,2;
```

website_session_id	product_page_seen	cart_page	shipping_page	billing2_page	thankyou_page
63513	/the-original-mr-fuzzy	1	0	0	0
63513	/the-original-mr-fuzzy	0	1	0	0
63513	/the-original-mr-fuzzy	0	0	1	0
63513	/the-original-mr-fuzzy	0	0	0	1
63515	/the-original-mr-fuzzy	1	0	0	0
63516	/the-original-mr-fuzzy	0	0	0	0
63517	/the-original-mr-fuzzy	0	0	0	0
63518	/the-original-mr-fuzzy	1	0	0	0
63519	/the-original-mr-fuzzy	0	0	0	0
63520	/the-original-mr-fuzzy	0	0	0	0
63521	/the-original-mr-fuzzy	1	0	0	0
63521	/the-original-mr-fuzzy	0	1	0	0
63521	/the-original-mr-fuzzy	0	0	1	0
63521	/the-original-mr-fuzzy	0	0	0	1
63526	/the-original-mr-fuzzy	0	0	0	0

```
create temporary table session_product_label_made_it_flag
select
        website_session_id,
    case
            when product_page_seen = '/the-original-mr-fuzzy' then 'mfsuzzy'
        when product_page_seen = '/the-forever-love-bear' then 'lovebear'
        else 'check logic'
        end as product_seen,
    max(cart_page) as cart_made_it,
    max(shipping_page) as shipping_made_it,
    max(billing2_page) as billing2_made_it,
    max(thankyou_page) as thankyou_made_it
from (
        select
            session_seeing_products_pages.website_session_id,
            session_seeing_products_pages.product_page_seen,
            case when pageview_url = '/cart' then 1 else 0 end as cart_page,
            case when pageview_url = '/shipping' then 1 else 0 end as shipping_page,
            case when pageview_url = '/billing-2' then 1 else 0 end as billing2_page,
            case when pageview_url = '/thank-you-for-your-order' then 1 else 0 end as thankyou_page
        from session_seeing_products_pages
            left join website_pageviews
```

```
                    on website_pageviews.website_session_id =
session_seeing_products_pages.website_session_id
                    and website_pageviews.website_pageview_id >
session_seeing_products_pages.website_pageview_id
        order by 1,2) as pageview_label
   group by 1,2;
```

website_session_id	product_seen	cart_made_it	shipping_made_it	billing2_made_it	thankyou_made_it
63513	mrfuzzy	1	1	1	1
63515	mrfuzzy	1	0	0	0
63516	mrfuzzy	0	0	0	0
63517	mrfuzzy	0	0	0	0
63518	mrfuzzy	1	0	0	0
63519	mrfuzzy	0	0	0	0
63520	mrfuzzy	0	0	0	0
63521	mrfuzzy	1	1	1	1
63526	mrfuzzy	0	0	0	0
63527	mrfuzzy	0	0	0	0
63528	mrfuzzy	0	0	0	0
63529	mrfuzzy	0	0	0	0
63532	mrfuzzy	1	0	0	0
63533	mrfuzzy	0	0	0	0
63534	mrfuzzy	1	1	1	1

•••

```
                    -- step-4: create session-level convertion funnel view
select
        count(distinct website_session_id) as sessions,
   count(distinct case when cart_made_it = 1 then website_session_id else null end) as to_cart,
   count(distinct case when cart_made_it = 1 then website_session_id else null end) /
     count(distinct website_session_id) as cart_click_rt,
   count(distinct case when shipping_made_it = 1 then website_session_id else null end) as to_shipping,
   count(distinct case when shipping_made_it = 1 then website_session_id else null end) /
     count(distinct website_session_id) as shipping_click_rt,
   count(distinct case when billing2_made_it = 1 then website_session_id else null end) as to_billing2,
   count(distinct case when billing2_made_it = 1 then website_session_id else null end) /
     count(distinct website_session_id) as billing2_click_rt,
   count(distinct case when thankyou_made_it = 1 then website_session_id else null end) as to_thankyou,
   count(distinct case when thankyou_made_it = 1 then website_session_id else null end) /
     count(distinct website_session_id) as thankyou_click_rt
from session_product_label_made_it_flag
group by product_seen;
```

Findings:

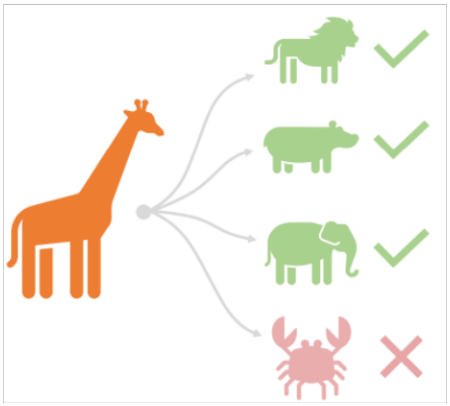
sessions	to_cart	cart_click_rt	to_shipping	shipping_click_rt	to_billing2	billing2_click_rt	to_thankyou	thankyou_click_rt
1599	877	0.5485	603	0.3771	488	0.3052	301	0.1882
6985	3038	0.4349	2084	0.2984	1710	0.2448	1088	0.1558

The findings are encouraging! Our initial discovery that adding a second product boosted overall clickthrough rates from the /products page is validated by this analysis. Specifically, it reveals that the Love Bear exhibits a better click rate to the /cart page and comparable rates throughout the rest of the conversion funnel. It strongly indicates that the addition of the

second product, Love Bear, has been a highly beneficial and successful expansion for our business.

What Is Cross-Selling Products?

Cross-sell analysis involves gaining insights into which products users are most likely to purchase together and leveraging this information to offer strategic product recommendations. This process includes identifying products frequently bought together, testing and optimizing cross-selling strategies on your website, and understanding the impact on conversion rates and overall revenue when attempting to cross-sell additional products. In essence, cross-sell analysis aims to enhance the effectiveness of product recommendations and drive additional revenue by encouraging complementary purchases.



COMMON USE CASES:

- Understanding which products are often purchased together
- Testing and optimizing the way you cross-sell products on your website
- Understanding the conversion rate impact and the overall revenue impact of trying to cross sell additional products

CROSS-SELL ANALYSIS

November 22, 2013

Can you compare the metrics for the month before and the month after we introduced the option for customers to add a second product on the /cart page starting September 25th? Specifically, I'd like to see the clickthrough rate (CTR) from the /cart page, average products per order, average order value (AOV), and overall revenue per /cart page view. This analysis will provide insights into the impact of the change on customer behavior and overall revenue.

SQL QUERY:

```
-- step-1: identify the relivent /cart pageview and their sessions
create temporary table session_seeing_cart
select
    case
```

```
            when created_at < '2013-09-25' then 'A.Pre_cross_sell'
        when created_at > '2013-01-06' then 'B.Post_cross_sell'
        else 'check logic'
            end as time_period,
            website_session_id as cart_session_id,
            website_pageview_id as cart_pageview_id
from website_pageviews
where created_at between '2013-08-25' and '2013-10-25'
            and pageview_url = '/cart';
```

time_period	cart_session_id	cart_pageview_id
A.Pre_cross_sell	122947	285248
A.Pre_cross_sell	122952	285261
A.Pre_cross_sell	122954	285269
A.Pre_cross_sell	122963	285290
A.Pre_cross_sell	122973	285308
A.Pre_cross_sell	122974	285315
A.Pre_cross_sell	122987	285340
A.Pre_cross_sell	123008	285376
A.Pre_cross_sell	123016	285390
A.Pre_cross_sell	123020	285403
A.Pre_cross_sell	123021	285408
A.Pre_cross_sell	123030	285428
A.Pre_cross_sell	123031	285440
A.Pre_cross_sell	123033	285445
A.Pre_cross_sell	123038	285453

•••

```
            -- step-2: seeing which of the /cart session click through the sheeping page
create temporary table cart_sessions_seeing_anather_page
select
            session_seeing_cart.time_period,
    session_seeing_cart.cart_session_id,
    min(website_pageviews.website_pageview_id) as pv_id_after_cart
from session_seeing_cart
            left join website_pageviews
                        on website_pageviews.website_session_id = session_seeing_cart.cart_session_id
        and website_pageviews.website_pageview_id > session_seeing_cart.cart_session_id
group by 1,2
having 3 is not null;
```

time_period	cart_session_id	pv_id_after_cart
A.Pre_cross_sell	122947	285245
A.Pre_cross_sell	122952	285258
A.Pre_cross_sell	122954	285266
A.Pre_cross_sell	122963	285287
A.Pre_cross_sell	122973	285305
A.Pre_cross_sell	122974	285309
A.Pre_cross_sell	122987	285331
A.Pre_cross_sell	123008	285369
A.Pre_cross_sell	123016	285387
A.Pre_cross_sell	123020	285398
A.Pre_cross_sell	123021	285400
A.Pre_cross_sell	123030	285425
A.Pre_cross_sell	123031	285431
A.Pre_cross_sell	123033	285434
A.Pre_cross_sell	123038	285450

•••

```
            -- step-3: find the order associated with /cart session. Analyzed product purchesed, AOV
create temporary table pre_post_session_order
```

```
select
        time_period,
    cart_session_id,
    order_id,
    items_purchased,
    price_usd
from session_seeing_cart
        inner join orders
                on session_seeing_cart.cart_session_id = orders.website_session_id;
```

time_period	cart_session_id	order_id	items_purchased	price_usd
A.Pre_cross_sell	122947	6645	1	49.99
A.Pre_cross_sell	122987	6646	1	49.99
A.Pre_cross_sell	123008	6647	1	49.99
A.Pre_cross_sell	123021	6648	1	59.99
A.Pre_cross_sell	123030	6649	1	49.99
A.Pre_cross_sell	123031	6650	1	49.99
A.Pre_cross_sell	123038	6651	1	49.99
A.Pre_cross_sell	123049	6652	1	49.99
A.Pre_cross_sell	123055	6653	1	49.99
A.Pre_cross_sell	123062	6654	1	49.99
A.Pre_cross_sell	123065	6655	1	49.99
A.Pre_cross_sell	123075	6656	1	49.99
A.Pre_cross_sell	123081	6657	1	49.99
A.Pre_cross_sell	123101	6658	1	59.99
A.Pre_cross_sell	123103	6659	1	49.99

•••

```
select
        session_seeing_cart.time_period,
    session_seeing_cart.cart_session_id,
    case when cart_sessions_seeing_anather_page.cart_session_id is null then 0 else 1 end as
click_to_anather_page,
    case when pre_post_session_order.order_id is null then 0 else 1 end as place_order,
    pre_post_session_order.items_purchased,
    pre_post_session_order.price_usd
from session_seeing_cart
        left join cart_sessions_seeing_anather_page
                on session_seeing_cart.cart_session_id = cart_sessions_seeing_anather_page.cart_session_id
        left join pre_post_session_order
                on session_seeing_cart.cart_session_id = pre_post_session_order.cart_session_id
order by 2;
```

time_period	cart_session_id	click_to_anather_page	place_order	items_purchased	price_usd
A.Pre_cross_sell	122947	1	1	1	49.99
A.Pre_cross_sell	122952	1	0	NULL	NULL
A.Pre_cross_sell	122954	1	0	NULL	NULL
A.Pre_cross_sell	122963	1	0	NULL	NULL
A.Pre_cross_sell	122973	1	0	NULL	NULL
A.Pre_cross_sell	122974	1	0	NULL	NULL
A.Pre_cross_sell	122987	1	1	1	49.99
A.Pre_cross_sell	123008	1	1	1	49.99
A.Pre_cross_sell	123016	1	0	NULL	NULL
A.Pre_cross_sell	123020	1	0	NULL	NULL
A.Pre_cross_sell	123021	1	1	1	59.99
A.Pre_cross_sell	123030	1	1	1	49.99
A.Pre_cross_sell	123031	1	1	1	49.99
A.Pre_cross_sell	123033	1	0	NULL	NULL
A.Pre_cross_sell	123038	1	1	1	49.99

•••

```
select
  time_period,
  count(distinct cart_session_id) as cart_sessions,
```

```
  sum(click_to_anather_page) as clickthroughs,
  sum(click_to_anather_page) / count(distinct cart_session_id) as cart_CTR,
  sum(place_order) as order_placed,
  sum(items_purchased) as product_purchased,
  sum(items_purchased) / sum(place_order) as product_pur_order,
  sum(price_usd) as revinue,
  sum(price_usd) / sum(place_order) as AOV,
  sum(price_usd) / count(distinct cart_session_id) as rev_per_cart_session
from (
select
        session_seeing_cart.time_period,
    session_seeing_cart.cart_session_id,
    case when cart_sessions_seeing_anather_page.cart_session_id is null then 0 else 1 end as
click_to_anather_page,
    case when pre_post_session_order.order_id is null then 0 else 1 end as place_order,
    pre_post_session_order.items_purchased,
    pre_post_session_order.price_usd
from session_seeing_cart
        left join cart_sessions_seeing_anather_page
                on session_seeing_cart.cart_session_id = cart_sessions_seeing_anather_page.cart_session_id
        left join pre_post_session_order
                on session_seeing_cart.cart_session_id = pre_post_session_order.cart_session_id
order by 2)as full_data
group by time_period ;
```

Findings:

time_period	cart_sessions	clickthroughs	cart_CTR	order_placed	product_purchased	product_pur_order	revinue	AOV	rev_per_cart_session
A.Pre_cross_sell	1830	1830	1.0000	652	652	1.0000	33523.48	51.416380	18.318842
B.Post_cross_sell	1975	1975	1.0000	671	701	1.0447	36402.99	54.251848	18.431894

The analysis reveals reassuring results – the clickthrough rate (CTR) from the /cart page did not decrease, alleviating initial concerns. Moreover, there are positive trends, with slight improvements observed in average products per order, average order value (AOV), and revenue per /cart session since the introduction of the cross-sell feature. While it may not be a game-changer, the overall trend indicates positive customer engagement and increased revenue potential with the addition of the cross-sell option.

PORTFOLIO EXPANSION ANALYSIS

January 12, 2014

Can you conduct a pre-post analysis comparing the month before and the month after the launch of our third product, Birthday Bear, on December 12th, 2013? Specifically, I'd like to assess changes in session-to-order conversion rate, average order value (AOV), products per order, and revenue per session. This analysis will provide insights into the impact of the new product on customer behavior and overall performance metrics.

SQL QUERY:

```
SELECT
CASE
    WHEN website_sessions.created_at < '2013-12-12' THEN 'A_Pre-Birthday_Bear'
    WHEN website_sessions.created_at >= '2013-12-12' THEN 'B_Post_Birthday_Bear'
    ELSE 'uh oh...check logic'
END AS time_period,
COUNT(DISTINCT website_sessions.website_session_id) AS sessions,
COUNT(DISTINCT orders.order_id) AS orders,
COUNT(DISTINCT orders.order_id)/COUNT(DISTINCT website_sessions.website_session_id) AS conv_rate,
SUM(orders.price_usd) AS total_revenue,
SUM(orders.price_usd)/COUNT(DISTINCT orders.order_id) as average_order_value,
SUM(orders.items_purchased)/COUNT(DISTINCT orders.order_id) as products_per_order,
SUM(orders.items_purchased)/COUNT(DISTINCT website_sessions.website_session_id)  AS
revenue_per_session

FROM website_sessions

LEFT JOIN orders
ON orders.website_session_id = website_sessions.website_session_id

WHERE website_sessions.created_at BETWEEN '2013-11-12' AND '2014-01-12'

GROUP BY 1;
```

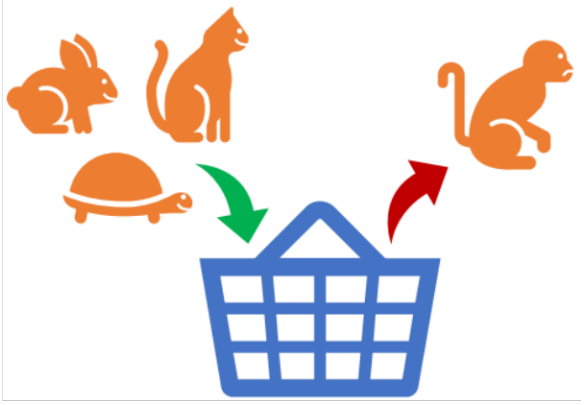
Findings:

time_period	sessions	orders	conv_rate	total_revenue	average_order_value	products_per_order	revenue_per_session
A_Pre-Birthday_Bear	17343	1055	0.0608	57208.96	54.226502	1.0464	0.0637
B_Post_Birthday_Bear	13383	940	0.0702	53515.44	56.931319	1.1234	0.0789

The analysis indicates that all our critical metrics have shown improvement since the launch of the third product. This is indeed fantastic news, highlighting the positive impact of the new product on session-to-order conversion rate, average order value (AOV), products per order, and revenue per session.

What Is Product Refund Analysis?

Analyzing product refund rates is essential for quality control and identifying areas that may require attention. This process involves monitoring products from various suppliers, understanding refund rates based on different price points, and incorporating product refund rates and associated costs into the overall assessment of business performance. By doing so, businesses can maintain quality standards, pinpoint potential issues, and make informed decisions about product offerings and supplier relationships.



COMMON USE CASES:

- Monitoring products from different suppliers
- Understanding refund rates for products at different price points
- Taking product refund rates and the associated costs into account when assessing the overall performance of your business

PRODUCT REFUND RATES

October 15, 2014

Can you retrieve and present the monthly product refund rates, broken down by product, particularly focusing on Mr. Fuzzy, from September 2013 to the present? This analysis will help confirm whether the quality issues with Mr. Fuzzy, which persisted until September 2013 and resurfaced in Aug/Sep 2014, have been effectively addressed since we replaced the supplier on September 16, 2014.

SQL QUERY:

```
SELECT
  YEAR(order_items.created_at) AS yr,
  MONTH(order_items.created_at) AS mo,
COUNT(DISTINCT CASE WHEN product_id = 1 THEN order_items.order_item_id ELSE NULL END) AS
p1_orders,
  COUNT(DISTINCT CASE WHEN product_id = 1 THEN order_item_refunds.order_item_id ELSE NULL END) /
    COUNT(DISTINCT CASE WHEN product_id = 1 THEN order_items.order_item_id ELSE NULL END) as
P1_refund_rt,

COUNT(DISTINCT CASE WHEN product_id = 2 THEN order_items.order_item_id ELSE NULL END) AS
p2_orders,
  COUNT(DISTINCT CASE WHEN product_id = 2 THEN order_item_refunds.order_item_id ELSE NULL END) /
    COUNT(DISTINCT CASE WHEN product_id = 2 THEN order_items.order_item_id ELSE NULL END) as
P2_refund_rt,

COUNT(DISTINCT CASE WHEN product_id = 3 THEN order_items.order_item_id ELSE NULL END) AS
p3_orders,
  COUNT(DISTINCT CASE WHEN product_id = 3 THEN order_item_refunds.order_item_id ELSE NULL END) /
```

```
    COUNT(DISTINCT CASE WHEN product_id = 3 THEN order_items.order_item_id ELSE NULL END) as
P3_refund_rt,

COUNT(DISTINCT CASE WHEN product_id = 4 THEN order_items.order_item_id ELSE NULL END) AS
p4_orders,
  COUNT(DISTINCT CASE WHEN product_id = 4 THEN order_item_refunds.order_item_id ELSE NULL END) /
    COUNT(DISTINCT CASE WHEN product_id = 4 THEN order_items.order_item_id ELSE NULL END) as
P4_refund_rt
FROM order_items
  LEFT JOIN order_item_refunds
    ON order_items.order_item_id = order_item_refunds.order_item_id
WHERE order_items.created_at < '2024-12-15'
GROUP BY 1, 2;
```

Findings:

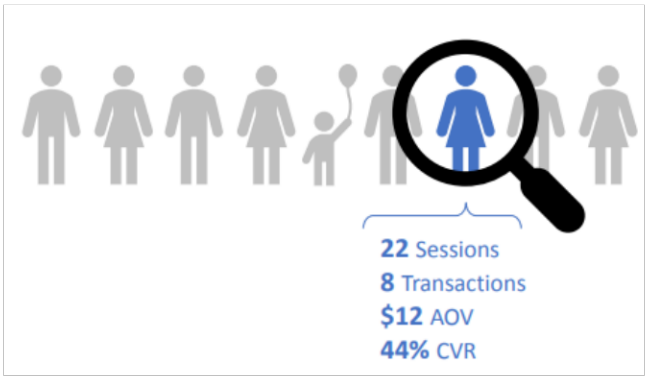
yr	mo	p1_orders	P1_refund_rt	p2_orders	P2_refund_rt	p3_orders	P3_refund_rt	p4_orders	P4_refund_rt
2012	6	140	0.0571	0	NULL	0	NULL	0	NULL
2012	7	169	0.0828	0	NULL	0	NULL	0	NULL
2012	8	228	0.0746	0	NULL	0	NULL	0	NULL
2012	9	287	0.0906	0	NULL	0	NULL	0	NULL
2012	10	371	0.0728	0	NULL	0	NULL	0	NULL
2012	11	618	0.0744	0	NULL	0	NULL	0	NULL
2012	12	506	0.0593	0	NULL	0	NULL	0	NULL
2013	1	343	0.0496	47	0.0213	0	NULL	0	NULL
2013	2	336	0.0714	162	0.0123	0	NULL	0	NULL
2013	3	320	0.0563	65	0.0462	0	NULL	0	NULL
2013	4	459	0.0414	94	0.0106	0	NULL	0	NULL
2013	5	489	0.0634	82	0.0244	0	NULL	0	NULL
2013	6	503	0.0775	90	0.0556	0	NULL	0	NULL
2013	7	509	0.0727	95	0.0316	0	NULL	0	NULL
2013	8	510	0.0549	98	0.0102	0	NULL	0	NULL
2013	9	537	0.0428	98	0.0102	0	NULL	0	NULL
2013	10	603	0.0282	135	0.0148	0	NULL	0	NULL
2013	11	724	0.0345	174	0.0230	0	NULL	0	NULL
2013	12	818	0.0232	183	0.0219	139	0.0719	0	NULL
2014	1	728	0.0426	183	0.0219	200	0.0650	0	NULL

The analysis reveals that the refund rates for Mr. Fuzzy did decrease following the initial improvements in September 2013. However, as expected, refund rates spiked to 13-14% in August and September due to the major quality issues during that period. On a positive note, it seems that the new supplier, implemented on September 16, 2014, has significantly improved the situation, showing much better performance. Additionally, the refund rates for other products appear to be satisfactory.

USER ANALYSIS:

Analyzing repeat visits is crucial for understanding user behavior and recognizing the significance of valuable customers. This process involves examining repeat activity to determine how frequently customers return to your site, understanding the channels they use during return

visits, and evaluating whether you are incurring additional costs for their return through paid channels. By leveraging insights from repeat visit activity, businesses can enhance their understanding of the customer's value, facilitating better optimization of marketing channels to maximize overall efficiency and effectiveness.



COMMON USE CASES:

- Analyzing repeat activity to see how often customers are coming back to visit your site
- Understanding which channels they use when they come back, and whether or not you are paying for them again through paid channels
- Using your repeat visit activity to build a better understanding of the value of a customer to better optimize marketing channels

Tracking Repeat Customers Across Multiple Sessions

Businesses employ browser cookies to track customer behavior across multiple sessions. These cookies are equipped with unique ID values that enable the recognition of a customer when they revisit the website, allowing businesses to monitor and analyze their behavior over time. This tracking mechanism provides valuable insights into customer preferences, interactions, and engagement, facilitating more personalized and targeted marketing efforts.

website_session_id	created_at	user_id	is_repeat_session	utm_source	utm_campaign	utm_content	device_type	http_referer
237966	2014-04-28 23:02:55	204524	0	gsearch	nonbrand	g_ad_1	desktop	https://www.gsearch.com
319940	2014-09-11 12:38:39	271374	0	bsearch	nonbrand	b_ad_1	desktop	https://www.bsearch.com
326645	2014-09-22 13:50:39	271374	1	gsearch	brand	g_ad_2	desktop	https://www.gsearch.com
325116	2014-09-19 11:42:44	275579	0	socialbook	desktop_targeted	social_ad_2	desktop	https://www.socialbook.com
349691	2014-10-26 19:24:17	275579	1	gsearch	brand	g_ad_2	desktop	https://www.gsearch.com
357769	2014-11-06 22:10:25	275579	1	NULL	NULL	NULL	mobile	https://www.gsearch.com
367395	2014-11-19 14:56:33	275579	1	NULL	NULL	NULL	mobile	NULL

IDENTIFYING REPEAT VISITORS

November 01, 2014

Can you retrieve data on the number of website visitors who have returned for another session in 2014 to date? This information will help us reassess customer value, especially for those with

repeat sessions, potentially allowing us to allocate a slightly higher budget for customer acquisition if their value proves to be greater than initially thought.

SQL QUERY:

```
create temporary table session_w_repeats
select
        new_sessions.user_id,
    new_sessions.website_session_id as new_session_id,
    website_sessions.website_session_id as repeat_session_id
from (
select
        user_id,
    website_session_id
from website_sessions
where created_at < '2014-11-01'      -- from
        and created_at >= '2014-01-01'   -- asked date
    and is_repeat_session = 0        -- new sessions only
) as new_sessions
        left join website_sessions
                on  website_sessions.user_id = new_sessions.user_id
        and website_sessions.is_repeat_session = 1           -- was a repeat session
        and website_sessions.website_session_id > new_sessions.website_session_id  -- session was leater then
new sessions
        and website_sessions.created_at < '2014-11-01'
            and website_sessions.created_at >= '2014-01-01';
```

user_id	new_session_id	repeat_session_id
152826	175252	NULL
152827	175253	NULL
152828	175254	NULL
152829	175256	NULL
152830	175257	NULL
152831	175258	NULL
152832	175259	NULL
152833	175260	NULL
152834	175261	NULL
152835	175262	NULL
152836	175263	NULL
152837	175264	199112
152838	175266	NULL
152839	175267	NULL
152840	175269	NULL
152841	175271	NULL
152842	175272	NULL
152843	175273	NULL
152844	175274	NULL
152845	175275	NULL

...

```
select
        repeat_session,
    count(distinct user_id) as users
from (
        select
                user_id,
```

```
        count(distinct new_session_id) as new_session,
        count(distinct repeat_session_id) as repeat_session
    from session_w_repeats
    group by 1
    order by 3 desc
    ) as user_lavel
group by 1;
```

Findings:

repeat_session	users
0	126813
1	14086
2	315
3	4686

The analysis indicates that a significant number of our customers do return to our site after their initial session. This suggests a notable level of engagement and repeat interest among our website visitors, emphasizing the potential value of nurturing relationships with these returning customers.

ANALYZING REPEAT BEHAVIOR

November 03, 2014

Can you assist me in understanding the behavior of our repeat customers by providing data on the minimum, maximum, and average time between the first and second session? I'm particularly interested in analyzing the time period from the beginning of 2014 to date. This information will offer insights into the frequency and patterns of engagement among customers who return for multiple sessions.

SQL QUERY:

```
   -- step-1: Identify the relevant new session and use the userID values to find any repeated sessions those
                                                    user has
create temporary table session_w_repeats_for_time_difference
select
        new_sessions.user_id,
   new_sessions.website_session_id as new_session_id,
   new_sessions.created_at as new_session_created_at,
   website_sessions.website_session_id as repeat_session_id,
   website_sessions.created_at as repeat_session_created_at
from (
select
        user_id,
   website_session_id,
   created_at
from website_sessions
where created_at < '2014-11-03'      -- from
        and created_at >= '2014-01-01'   -- asked date
```

```
    and is_repeat_session = 0        -- new sessions only
) as new_sessions
        left join website_sessions
                on  website_sessions.user_id = new_sessions.user_id
        and website_sessions.is_repeat_session = 1          -- was a repeat session
        and website_sessions.website_session_id > new_sessions.website_session_id  -- session was leater then
new sessions
        and website_sessions.created_at < '2014-11-03'
            and website_sessions.created_at >= '2014-01-01';
```

user_id	new_session_id	new_session_created_at	repeat_session_id	repeat_session_created_at
152826	175252	2014-01-01 00:00:09	NULL	NULL
152827	175253	2014-01-01 00:03:26	NULL	NULL
152828	175254	2014-01-01 00:26:21	NULL	NULL
152829	175256	2014-01-01 00:37:35	NULL	NULL
152830	175257	2014-01-01 00:41:11	NULL	NULL
152831	175258	2014-01-01 00:44:49	NULL	NULL
152832	175259	2014-01-01 00:48:18	NULL	NULL
152833	175260	2014-01-01 00:57:47	NULL	NULL
152834	175261	2014-01-01 01:02:43	NULL	NULL
152835	175262	2014-01-01 01:13:57	NULL	NULL
152836	175263	2014-01-01 01:29:35	NULL	NULL
152837	175264	2014-01-01 01:36:26	199112	2014-02-16 18:46:36
152838	175266	2014-01-01 01:50:54	NULL	NULL
152839	175267	2014-01-01 01:59:37	NULL	NULL
152840	175269	2014-01-01 02:04:15	NULL	NULL
152841	175271	2014-01-01 02:36:13	NULL	NULL
152842	175272	2014-01-01 02:39:48	NULL	NULL
152843	175273	2014-01-01 02:42:03	NULL	NULL
152844	175274	2014-01-01 02:45:36	NULL	NULL
152845	175275	2014-01-01 02:45:36	NULL	NULL

...

```
 -- step-2: find the created at time for the first and second sessions and find the difference between first and
                                                second sessions at a ser level
create temporary table user_first_to_second
select
        user_id,
   datediff(second_session_created_at, new_session_created_at) as day_first_to_second_session
from(
        select
                user_id,
                new_session_id,
                new_session_created_at,
                min(repeat_session_id) as second_session_id,
                min(repeat_session_created_at) as second_session_created_at
        from session_w_repeats_for_time_difference
        where repeat_session_id is not null
        group by 1,2,3
        ) as first_second;
```

user_id	day_first_to_second_session
152837	46
152847	28
152848	56
152849	15
152851	23
152865	49
152878	32
152881	64
152890	35
152891	40
152898	9
152901	41
152911	34
152932	40
152949	63
152953	30
152957	16
152962	19
152973	42
152988	65

...

```
-- step-3: aggrigate the user lavel data to find the avarage, min and max
select
    avg(day_first_to_second_session) as avg_day_first_to_second_session,
  min(day_first_to_second_session) as min_day_first_to_second_session,
  max(day_first_to_second_session) as max_day_first_to_second_session
from user_first_to_second;
```

Findings:

avg_day_first_to_second_session	min_day_first_to_second_session	max_day_first_to_second_session
33.2622	1	69

The analysis reveals that, on average, our repeat visitors return about a month later. This finding prompts the idea to investigate the channels through which these visitors are accessing the site. Understanding the source of their return visits can provide valuable insights into the effectiveness of various marketing channels and aid in optimizing strategies to engage repeat customers more effectively.

NEW VS REPEAT CHANNEL PATTERNS

November 05, 2014
Can you assist me in understanding the channels through which our repeat customers return? I'm curious to know if it's primarily direct type-in or if we are acquiring these customers through paid search ads multiple times. It would be highly valuable to compare new vs. repeat sessions by channel. If possible, could you pull this data for the year 2014 to date?

SQL QUERY:

```
select
    utm_source,
```

```
    utm_campaign,
    http_referer,
    count(case when is_repeat_session = 0 then website_session_id else null end) as new_session,
    count(case when is_repeat_session = 1 then website_session_id else null end) as repeat_session
from website_sessions
where created_at < '2014-11-05'
        and created_at >= '2014-01-01'
group by 1,2,3
order by 5 desc;
```

utm_source	utm_campaign	http_referer	new_session	repeat_session
NULL	NULL	NULL	6591	10564
NULL	NULL	https://www.gsearch.com	5738	9450
gsearch	brand	https://www.gsearch.com	5196	8897
bsearch	brand	https://www.bsearch.com	1236	2130
NULL	NULL	https://www.bsearch.com	1401	2057
gsearch	nonbrand	https://www.gsearch.com	100278	0
bsearch	nonbrand	https://www.bsearch.com	19672	0
socialbook	pilot	https://www.socialbook.com	5095	0
socialbook	desktop_targeted	https://www.socialbook.com	2557	0

```
select
        case
                when utm_source is null and http_referer in
('https://www.gsearch.com','https://www.bsearch.com') then 'organic search'
      when utm_campaign = 'nonbrand' then 'paid nonbrand'
      when utm_campaign = 'brand' then 'paid brand'
      when utm_source is null and http_referer is null then 'direct_type_in'
      when utm_source = 'socialbook' then 'paid social'
        end as channel_group,
                -- utm_source,
                -- utm_campaign,
                -- http_referer,
                count(case when is_repeat_session = 0 then website_session_id else null end) as
new_session,
                count(case when is_repeat_session = 1 then website_session_id else null end) as
repeat_session
from website_sessions
where created_at < '2014-11-05'
        and created_at >= '2014-01-01'
group by 1
order by repeat_session desc;
```

Findings:

channel_group	new_session	repeat_session
organic search	7139	11507
paid brand	6432	11027
direct_type_in	6591	10564
paid nonbrand	119950	0
paid social	7652	0

The analysis indicates that when customers return for repeat visits, they primarily come through organic search, direct type-in, and paid brand channels. Interestingly, only about one-third of these return visits are through a paid channel, and it's noteworthy that brand clicks within paid channels are less expensive than nonbrand clicks. Overall, the cost associated with these subsequent visits is relatively low, suggesting that we are not paying a significant amount for customers returning for repeat sessions.

NEW VS REPEAT PERFORMANCE

November 08, 2014
Can you help me compare the conversion rates and revenue per session for repeat sessions versus new sessions, using data from 2014 year to date? This comparison will provide insights into the performance differences between customers returning for repeat visits and those visiting for the first time.

SQL QUERY:

```
select
        is_repeat_session,
    count(distinct website_sessions.website_session_id) as sessions,
    count(distinct orders.order_id) / count(distinct website_sessions.website_session_id) as convertion_rt,
    sum(price_usd) / count(distinct website_sessions.website_session_id) as ravinue_per_session
from website_sessions
        left join orders
                on website_sessions.website_session_id = orders.website_session_id
where website_sessions.created_at < '2014-11-08'
        and website_sessions.created_at >= '2014-01-01'
group by 1;
```

Findings:

is_repeat_session	sessions	convertion_rt	ravinue_per_session
0	149787	0.0680	4.343754
1	33577	0.0811	5.168828

The analysis reveals intriguing insights – repeat sessions show a higher likelihood of conversion and generate more revenue per session compared to new sessions. Given that the cost associated with repeat sessions is relatively low, it suggests the importance of factoring them into our bidding strategy for paid traffic. I'll coordinate with Tom to explore ways to optimize our approach considering this valuable information.

NOTE:

The NIK Fuzzy Factory eCommerce Database Analysis project involves exploring and analyzing an eCommerce dataset for insights and informed decision-making. The dataset combines Kaggle data with additional contributions from the project creator, and ChatGPT was employed for data modification and creation to enhance its comprehensiveness.

Data Sources:

1. Kaggle:
    - Sourced a subset of diverse datasets from Kaggle, known for ML and data analysis competitions.
2. User-Contributed Data:
    - Project creator contributed supplementary files, potentially containing user interactions, feedback, and custom attributes specific to NIK Fuzzy Factory's eCommerce.
3. ChatGPT:
    - Utilized ChatGPT for data modification, creating synthetic data, simulating user behavior, and adding context to existing data points.