codecademy

CoolTShirts Capstone

Learn SQL from Scratch

Andrew Nygren

6/18/2018

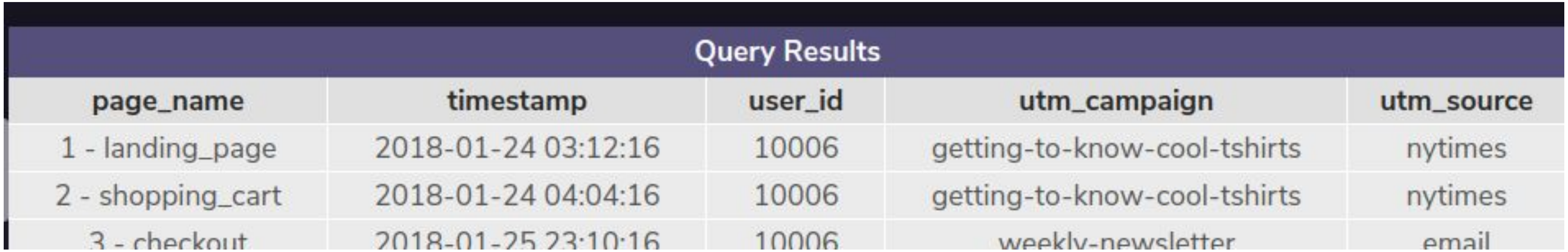
Table of Contents

1.1 Getting to know CoolTshirts - Basic Layout

Selecting * in a query will return all columns and rows from the page_visits table. Using this we can explore the The CoolTshits database.
Using the Limit function we can just return a few rows and find that we have 5 columns.

- page_name
- timestamp
- user_id
- utm_campaign
- utm_source

```
SELECT *
FROM page_visits
LIMIT 10;
```



Query Results

page_name	timestamp	user_id	utm_campaign	utm_source
1 - landing_page	2018-01-24 03:12:16	10006	getting-to-know-cool-tshirts	nytimes
2 - shopping_cart	2018-01-24 04:04:16	10006	getting-to-know-cool-tshirts	nytimes
3 - checkout	2018-01-25 23:10:16	10006	weekly-newsletter	email

1.2 Getting to know CoolTshirts - Number of Campaigns and Sources

The utm_campaign is the type of ad that is being served to the user. The utm_source is the method of delivery. A campaign may be used on multiple sources.

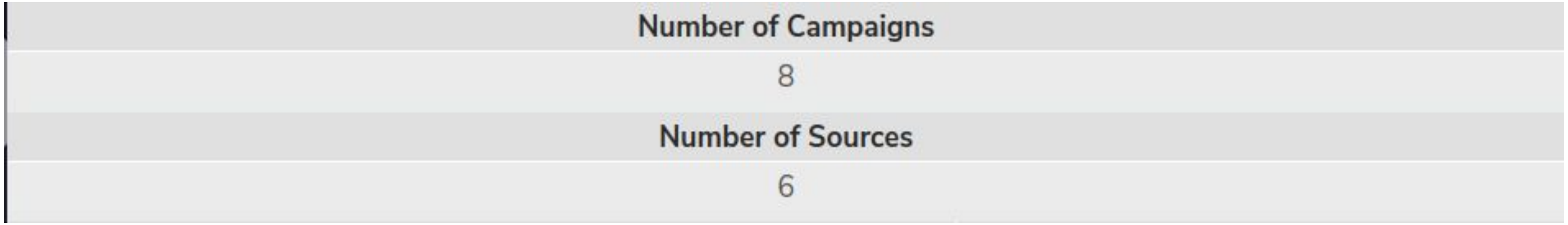
We can find how many campaigns we have with the following query:

```
SELECT COUNT(DISTINCT utm_campaign) AS 'Number of Campaigns'
FROM page_visits;
```

And we can find how many sources we have with the following query:

```
SELECT COUNT(DISTINCT utm_source) AS 'Number of Sources'
FROM page_visits;
```

The count function will count the number of rows returned and the distinct function will limit the rows to (1) of each campaign or source. Therefore we will return (8) rows of distinct campaign values and count those (8) rows or (6) rows of distinct source values and count those (6) rows both using AS to alias their headers



Number of Campaigns
8

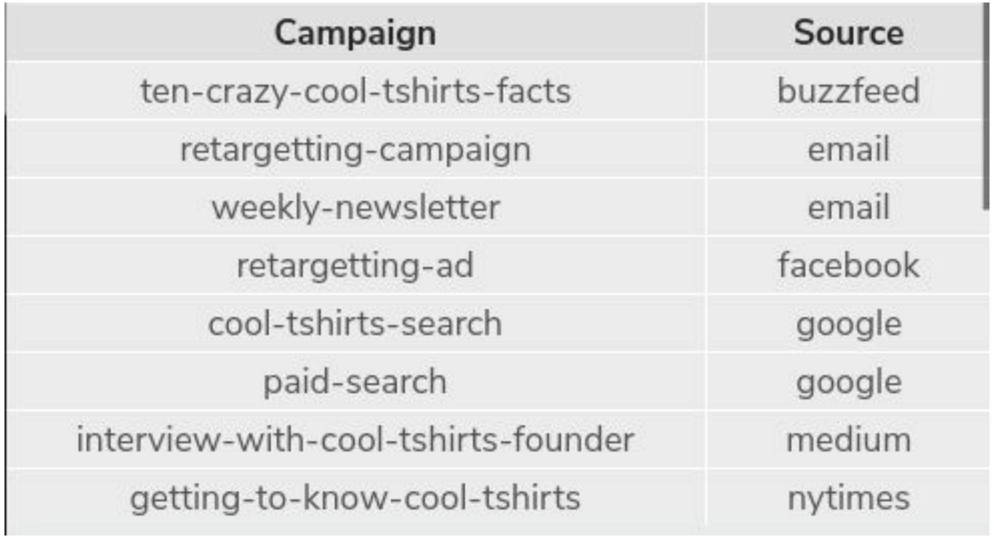
Number of Sources
6

1.3 Getting to know CoolTshirts - Campaigns Related to Sources

We will see which campaigns were used by what source .
My first inclination was to select the campaign and source then group by the campaign. It gave me the correct results for the data supplied, but I realized that I would lose data if a single campaign was used on multiple sources, I would only return the first source on the first instance of the campagn.

I thought that DISTINCT would do the same, but after testing I realized that it would return a distinct campaign for a distinct source. I ordered by the source so it would be sort of group together all the campaigns used on a single source

```
SELECT  DISTINCT utm_campaign AS
'Campaign',utm_source AS 'Source'
FROM page_visits
ORDER BY 2;
```

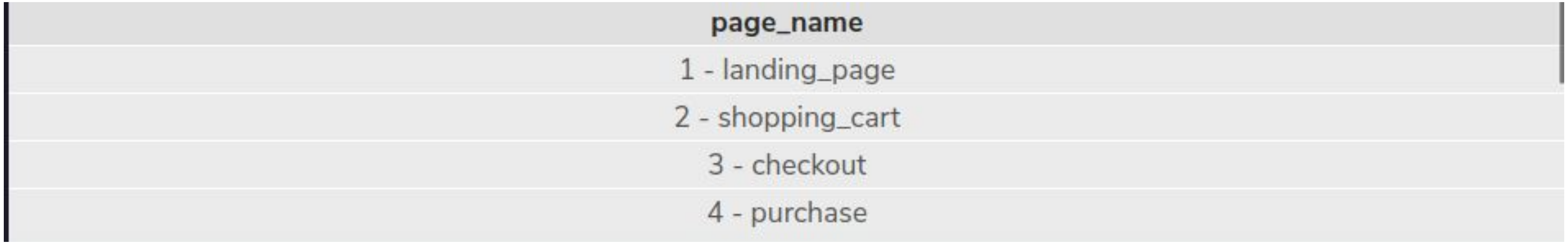


Campaign	Source
ten-crazy-cool-tshirts-facts	buzzfeed
retargetting-campaign	email
weekly-newsletter	email
retargetting-ad	facebook
cool-tshirts-search	google
paid-search	google
interview-with-cool-tshirts-founder	medium
getting-to-know-cool-tshirts	nytimes

1.4 Getting to know CoolTshirts - Pages

Lastly we will find the different page names by limiting the results using the distinct function which will return (1) of each page name. We will find that there are four (4) pages.

```
SELECT DISTINCT page_name
FROM page_visits;
```



page_name
1 - landing_page
2 - shopping_cart
3 - checkout
4 - purchase

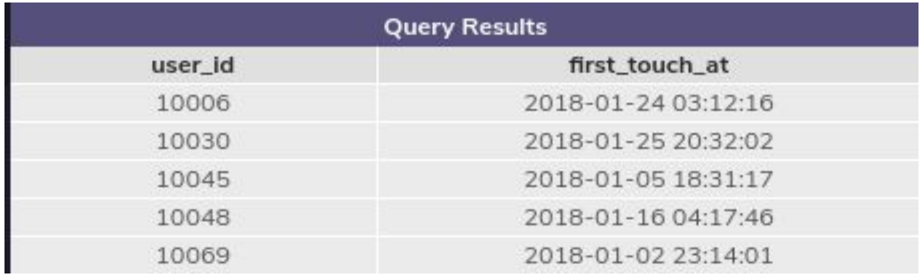
2.1 User journey - First touch

First touch will show us the first time someone visits the site. We can then use this to find out how they got there.

Using “WITH” we will create a subquery named “first_touch” we will select the user_id, and using “MIN” on the timestamp we will limit the results to the earliest time to be returned. Grouping by the user_id will limit the results to (1) of each user_id at the earliest time

```
WITH first_touch AS (
          SELECT user_id,
          MIN(timestamp) as first_touch_at
   FROM page_visits
   GROUP BY user_id)
```

This is an example of returned results by selecting all (*) from the first_touch subquery.

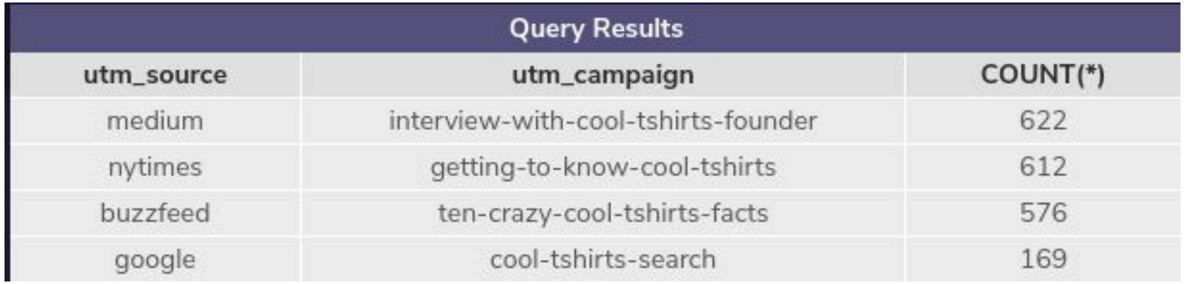


Query Results

user_id	first_touch_at
10006	2018-01-24 03:12:16
10030	2018-01-25 20:32:02
10045	2018-01-05 18:31:17
10048	2018-01-16 04:17:46
10069	2018-01-02 23:14:01

2.2 User journey - First touch Part 2

We can then use this subquery in our main query. We will select the utm_source and utm_ campaign and count the number of rows returned from the page_visits table. To limit this count to the first touch we will join the subquery to our page_visits table on the ft.user_id on pv.user_id and ft.first_touch_at on pv.timestamp. Grouping by the campaign will count up the rows returned per campaign of the joined tables. We ordered them by the count descending to see which campaigns were most successful.



Query Results

utm_source	utm_campaign	COUNT(*)
medium	interview-with-cool-tshirts-founder	622
nytimes	getting-to-know-cool-tshirts	612
buzzfeed	ten-crazy-cool-tshirts-facts	576
google	cool-tshirts-search	169

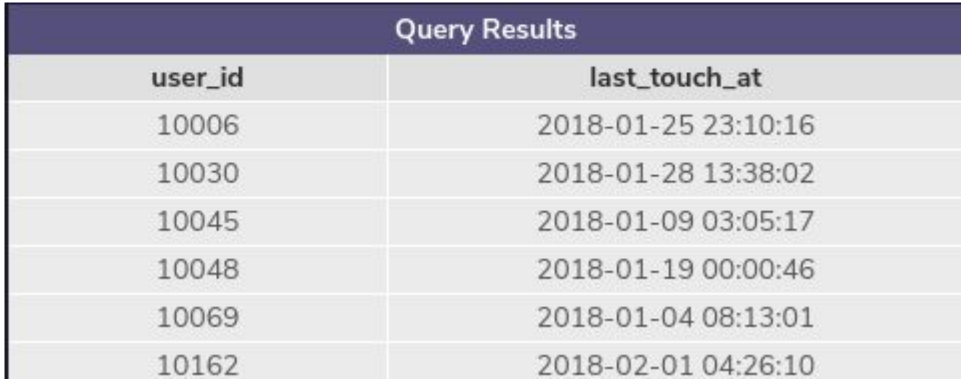
```
WITH first_touch AS (
   SELECT user_id,
      MIN(timestamp) as first_touch_at
   FROM page_visits
   GROUP BY user_id)
SELECT pv.utm_source,
      pv.utm_campaign,
      COUNT(*)
FROM first_touch ft
JOIN page_visits pv
   ON ft.user_id = pv.user_id
   AND ft.first_touch_at = pv.timestamp
   GROUP BY 2
   ORDER BY 3 DESC;
```

2.3 User journey - Last touch

To find the last touch count each campaign produced will be done in much the same way first touch campaign count was done.
Instead of using "MIN" to find the earliest time we will use "MAX" on the timestamp to limit the results to the latest time to be returned. Grouping by the user_id will limit the results to (1) of each user_id at the latest time.

```
WITH last_touch AS (
   SELECT user_id,
      MAX(timestamp) as last_touch_at
   FROM page_visits
   GROUP BY user_id)
```

This is an example of returned results by selecting all (*) from the last_touch table.



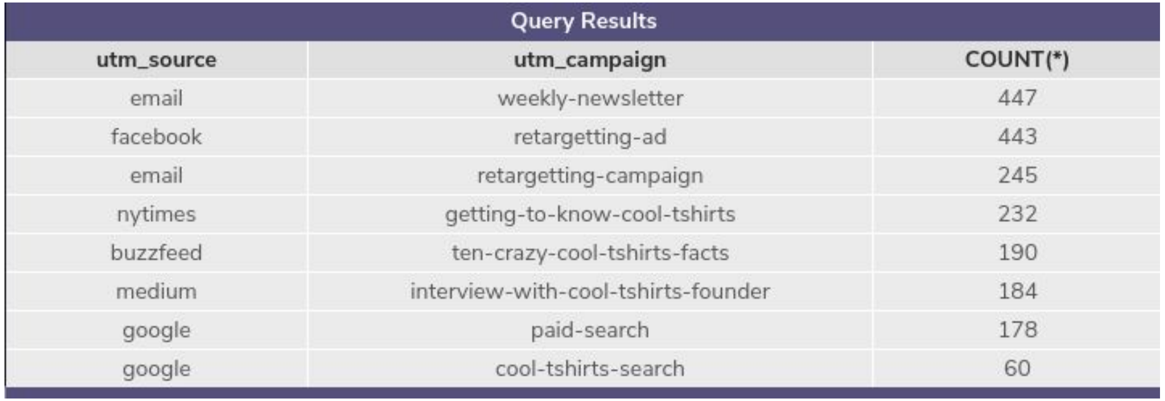
Query Results

user_id	last_touch_at
10006	2018-01-25 23:10:16
10030	2018-01-28 13:38:02
10045	2018-01-09 03:05:17
10048	2018-01-19 00:00:46
10069	2018-01-04 08:13:01
10162	2018-02-01 04:26:10

2.4 User journey - Last touch Part 2

As before We will join the subquery to the page_visits on the same points grouping and ordering the same way to achieve a total count per campaign.

```
WITH last_touch AS (
   SELECT user_id,
      MAX(timestamp) AS last_touch_at
   FROM page_visits
   GROUP BY user_id)
SELECT pv.utm_source,
      pv.utm_campaign,
      COUNT(*)
FROM last_touch lt
JOIN page_visits pv
   ON lt.user_id = pv.user_id
   AND lt.last_touch_at = pv.timestamp
   GROUP BY 2
   ORDER BY 3 DESC;
```



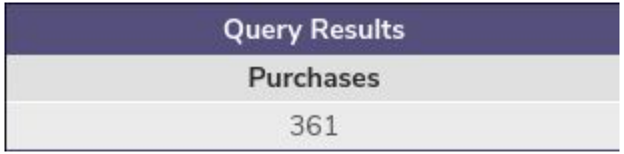
Query Results

utm_source	utm_campaign	COUNT(*)
email	weekly-newsletter	447
facebook	retargetting-ad	443
email	retargetting-campaign	245
nytimes	getting-to-know-cool-tshirts	232
buzzfeed	ten-crazy-cool-tshirts-facts	190
medium	interview-with-cool-tshirts-founder	184
google	paid-search	178
google	cool-tshirts-search	60

2.5 User journey - Total Purchases

Using COUNT we can select count everything aliasing as “Purchases”, from the page_visits table and then limit the count to the those who landed on the purchase page using a where clause.

```
SELECT COUNT(*) AS ‘Purchases’
FROM page_visits
WHERE page_name = '4 - purchase';
```



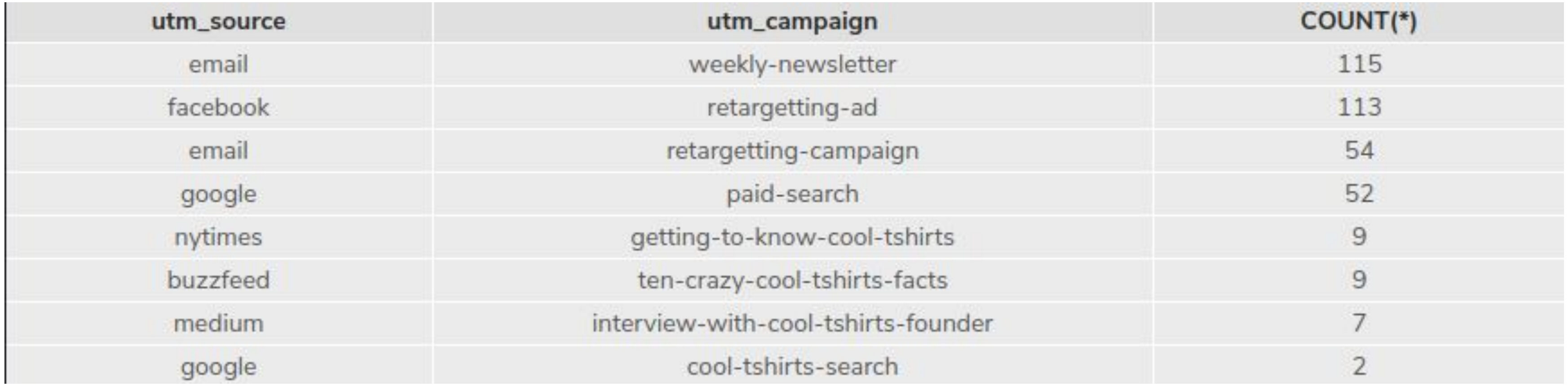
Query Results

Purchases
361

2.6 User journey - Last touch Purchases

Using our code from the last touch, we can add a where clause to our subquery to limit the returned results to those who made a purchase.

```
WITH last_touch AS (
  SELECT user_id,
         MAX(timestamp) AS last_touch_at
  FROM page_visits
  WHERE page_name = '4 - purchase'
  GROUP BY user_id)
SELECT pv.utm_source,
       pv.utm_campaign,
    COUNT(*)
FROM last_touch lt
JOIN page_visits pv
    ON lt.user_id = pv.user_id
    AND lt.last_touch_at = pv.timestamp
    GROUP BY 2
    ORDER BY 3 DESC;
```

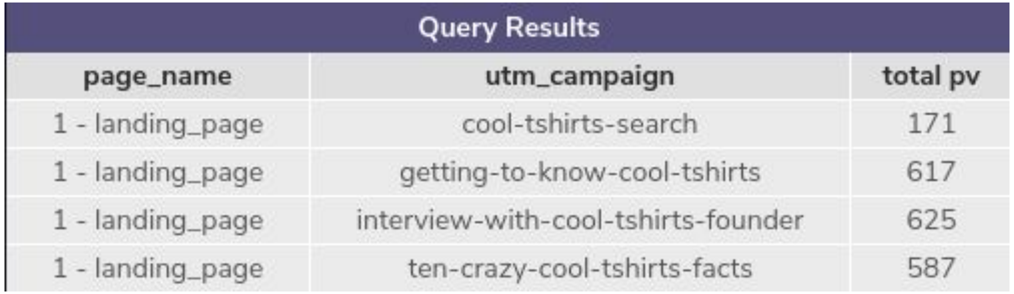


utm_source	utm_campaign	COUNT(*)
email	weekly-newsletter	115
facebook	retargetting-ad	113
email	retargetting-campaign	54
google	paid-search	52
nytimes	getting-to-know-cool-tshirts	9
buzzfeed	ten-crazy-cool-tshirts-facts	9
medium	interview-with-cool-tshirts-founder	7
google	cool-tshirts-search	2

2.7 User Journey - Typical User Journey

The typical user journey seems to me to be how most users get to the pages. So I decided to find out the MAX number of users a utm_campaign generated per page. I started out by making a subquery that would give the number of users visiting a particular page per campaign.

```
WITH fpv AS(
        SELECT page_name,
                utm_campaign,
                COUNT(*) AS 'total pv'
        FROM page_visits
        WHERE page_name like '1%'
        GROUP BY 2)
```



Query Results

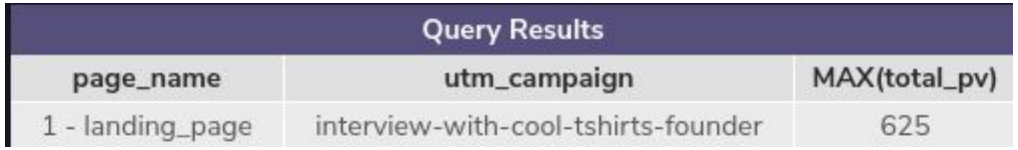
page_name	utm_campaign	total pv
1 - landing_page	cool-tshirts-search	171
1 - landing_page	getting-to-know-cool-tshirts	617
1 - landing_page	interview-with-cool-tshirts-founder	625
1 - landing_page	ten-crazy-cool-tshirts-facts	587

I made one of these for pages 1-4. I name the subquery for the page the counts are for, select the page name, campaign and count alias “total pv”. Taking the data from the page visits table, limit by the name, group by the campaign and count the totals.

2.8 User Journey - Typical User Journey Part 2

On the second part of the query I select the page name, campaign, and The MAX total pv. This will give us the largest number of those totals.

```
SELECT page_name,
             utm_campaign,
             MAX(total_pv)
FROM fpc
```

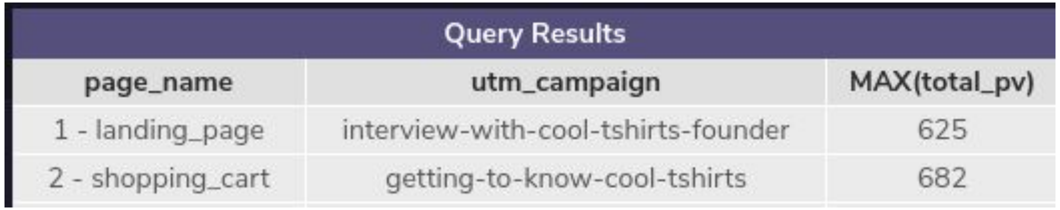


Query Results

page_name	utm_campaign	MAX(total_pv)
1 - landing_page	interview-with-cool-tshirts-founder	625

If we do this for each page we will get 4 seperate results. We can use UNION to bring these all together into one query.

```
SELECT page_name,
             utm_campaign,
             MAX(total_pv)
FROM fpv
UNION
SELECT page_name,
             utm_campaign,
             MAX(total_pv)
FROM spv
```

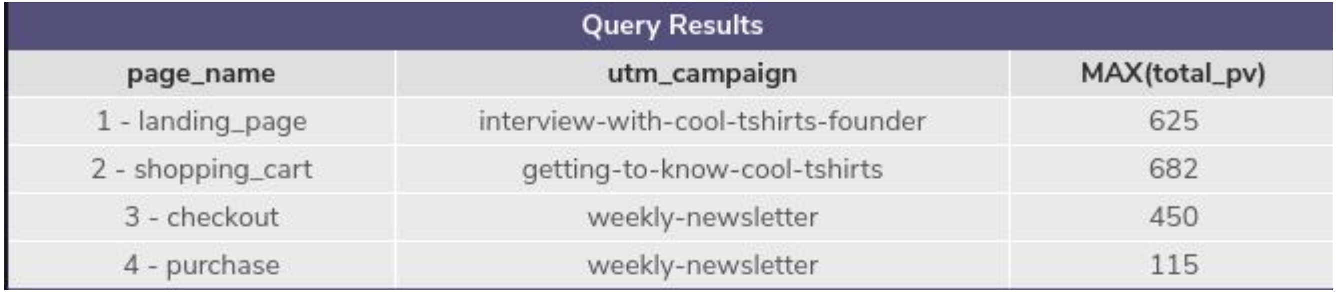


Query Results

page_name	utm_campaign	MAX(total_pv)
1 - landing_page	interview-with-cool-tshirts-founder	625
2 - shopping_cart	getting-to-know-cool-tshirts	682

2.9 User journey - Typical User journey Part 3

Bringing these all together we can see how most people landed on a page through what campaign.



Query Results

page_name	utm_campaign	MAX(total_pv)
1 - landing_page	interview-with-cool-tshirts-founder	625
2 - shopping_cart	getting-to-know-cool-tshirts	682
3 - checkout	weekly-newsletter	450
4 - purchase	weekly-newsletter	115

And so I conclude that a typical user journey starts with the campaign:

1. Interview with cool tshirts founder - to get to the landing page
2. Getting to know cool tshirts to get - to the shopping cart
3. Weekly Newsletter - to get to the checkout
4. Weekly Newsletter - to get to the purchase page

3.1 Optimize the campaign budget - Recommendations

If I were to recommend what campaigns to reinvest in, I would recommend the following:

1. Interview with cool tshirts founder - 622 first touches
2. Getting to know cool tshirts - 612 first touches
3. Ten crazy cool tshirt facts - 576 first touches
4. Weekly newsletter - 447 Last touches and 115 direct purchases
5. Retargeting ad - 443 Last touches and 113 direct purchases

I would recommend these because the had very high first touch and last touch results, driving the most traffic to the website, and most likely to result in a purchase.