

Dimensions (And custom dimensions!)

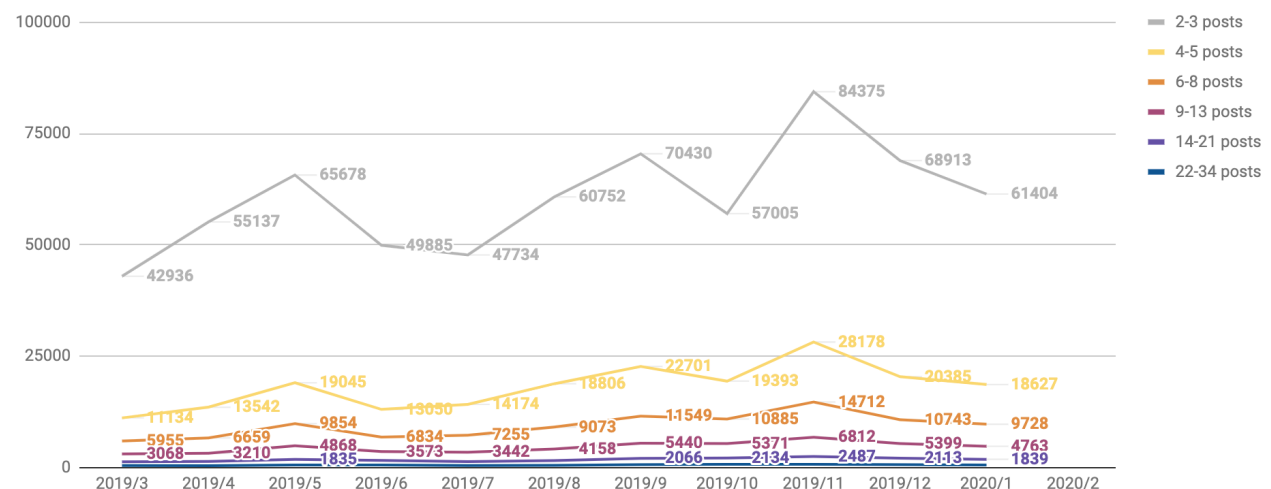
Understanding the behavior of our readers, subscribers and members

Brian Boyer, 2020/03/04

This will be useful to you if you are already using custom dimensions and want to automate reports that use them. It can also be helpful if you're new to dimensions and want to better understand the behavior of your audience. If you just wanna see the shiny reports, [scroll down to the Example reports](#) section at the end.

How many people read at least X posts/ in the last 30 days?

Values are cumulative. i.e. all 4-5 post readers are also counted as 2-3 post readers.



[Why implement custom dimensions?](#)

[What we're tracking](#)

[Contact Email](#)

[Contact Member Level](#)

[Contact 30-day reading](#)

[How to use custom dimension data](#)

[What articles did members read most?](#)

[When do users sign up for our newsletter?](#)

[How do our different users behave, compared to each other?](#)

[How we did it](#)

[Configuration in GA](#)

[What gets sent to GA?](#)

[Cookies!](#)

[But how do the cookies get set?](#)

[What about just asking folks to log in?](#)

[Example code \(Yes! Finally!\)](#)

[Writing the reading frequency data](#)

[Writing the email and membership data](#)

[Reading the cookie and sending the data to Google Analytics](#)

[Example reports!](#)

[The reading habits for different user segments](#)

[The number of people who read a lot of articles](#)

[Who are you? What's Memberkit?](#)

Why implement custom dimensions?

Do you ever wonder how different types of people interact with your site? What articles do younger people read? Do local readers behave differently than out-of-towners? What about your members? Do they read more geeky stuff? Or are they the same as everyone else?

Dimensions are the tool you'd use to answer questions like these. There are many dimensions built in to Google Analytics: the city a user is located in, the user's age and gender, the social network that referred a user, etc. (For a detailed list, check out the [Dimensions & Metrics Explorer](#).)

For example, here are the top pages on our site since the start of the year, first for users in Philadelphia...

Primary Dimension: Page Page Title Other ▼			
Plot Rows		Secondary dimension: City ▼	Sort Type: Default ▼
Advanced Filter O			
<input type="checkbox"/>	Page ?	City ?	Pageviews ? ↓
			543,766 % of Total: 41.16% (1,321,133)
<input type="checkbox"/>	1. /	Philadelphia	65,567 (12.06%)
<input type="checkbox"/>	2. /2020/02/12/10-great-black-owned-bars-to-check-out-in-philadelphia/	Philadelphia	19,238 (3.54%)
<input type="checkbox"/>	3. /2020/02/05/2-more-philly-cops-resign-before-firing-over-offensive-facebook-posts/	Philadelphia	14,938 (2.75%)
<input type="checkbox"/>	4. /2020/01/23/giant-aldi-another-trader-joes-all-the-new-supermarkets-planned-for-philly-this-year/	Philadelphia	12,613 (2.32%)

...and now for Pittsburgh.

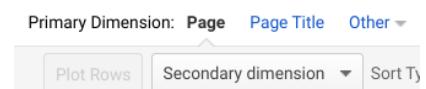
Primary Dimension: **Page** [Page Title](#) [Other](#) ▼

Plot Rows Secondary dimension: City ▼ Sort Type: Default ▼ **Advanced Filter**

<input type="checkbox"/>	Page ?	City ? ✕	Pageviews ? ↓
			5,639 % of Total: 0.43% (1,321,133)
<input type="checkbox"/>	1. /2015/12/31/pork-and-sauerkraut-on-new-years-day-why-the-pa-dutch-believe-its-your-luckiest-meal-of-the-year/	Pittsburgh	1,838 (32.59%)
<input type="checkbox"/>	2. /	Pittsburgh	294 (5.21%)
<input type="checkbox"/>	3. /2020/01/23/former-pa-lt-gov-mike-stack-hits-hollywood-with-new-standup-comedy-act/	Pittsburgh	143 (2.54%)
<input type="checkbox"/>	4. /2015/11/23/everything-you-dont-pay-sales-tax-on-in-pennsylvania-from-books-to-utilities/	Pittsburgh	140 (2.48%)

Looks like Philly users are into local lifestyle and hard news and Pittsburgh users are sharing great posts from our archives and reading state-wide news. Not surprising, since Billy Penn is a Philadelphia-focused news site and we don't expect many readers from Pittsburgh, but better to know than to assume!

(To look at this data on your site, go to Behavior > Site Content > All Pages. Then click the "Secondary dimension" dropdown and add "City". Finally, go to the Advanced Filter, select "Exactly matching" and type in the name of your city.)



So! Dimensions are useful! But what's a *custom* dimension, and why would you use one?

Custom dimensions are an additional layer of analytics that you can set up to track users by other things you know about them. You'll need a programmer on-hand to implement them, because you'll need to send additional data to Google Analytics in the Javascript that runs when your pages load.

We identified three custom dimensions that would be useful to our membership efforts: reading frequency (posts read in the last 30 days), membership status (member, non-member, etc), and email subscription (newsletters, etc) status.

By tracking those dimensions, we're able to answer valuable questions like...

- How often do newsletter subscribers visit our site?
- How many people read us a lot, and of those, how many don't get our newsletter?
- What articles do members read most? And what do our biggest donors read?

What we're tracking

To recap: In Google Analytics, dimensions are attributes of our users that let us slice our data. For example, their location, age, or what browser they're using. If you want to learn a lot more, read [Google's documentation on dimensions and metrics](#).

There are many dimensions built into GA. We've added several custom dimensions to track things that are specific to our user experience.

- Contact Email
- Contact Member Level
- Contact 30-day reading

Contact Email

When someone clicks a link in one of our emails, there's a little code attached at the end: https://denverite.com/2018/11/15/colorado-democrats-will-be-in-control-of-the-states-general-as-sembly-next-year-now-what/?mc_cid=f3d5611796&mc_eid=b31feb6b33. The code (in pink) can be used to identify the person to whom the email was sent. Our website uses that code to look up the contact's record in Mailchimp, and we write it down in a cookie that's stored by the user's browser. (Much more on how this works, [below](#).)

Then, as they click around the site, and on subsequent visits (even if they didn't visit via email), we tell Google Analytics a couple of things about the user. 1) What they're subscribed to, and 2) their member status. (No personally identifying information is sent to Google Analytics.)

Every user will have one of these three values for the *Contact Email* dimension:

Daily Newsletter	Users who subscribe to our daily newsletter.
Other	Users who are on our list, but aren't newsletter recipients. They might be subscribed to a story/yarn, an event attendee, etc.
Unknown	Users who we don't know anything about. They're

	probably not subscribers.
--	---------------------------

Contact Member Level

Every user will have one of these six values for the *Contact Member Level* dimension:

Member 3, Member 2, Member 1	Users who are members, and their membership level. Level 1 members give at least \$35/year, level 2 at least \$100, and level 3 at least \$500.
Donor	Users who have given us money, but below the threshold for membership. For example, someone giving us \$2/month, or someone who gave us a one-time donation of \$10.
None	Users who are on our list in Mailchimp, but who have not given any money.
Unknown	Users who we don't know anything about. They're probably not members.

A caveat about “Unknown” users

There are many reasons someone might be unknown to us, so the numbers we see in Google Analytics will never be perfectly accurate. For example...

- They don't get our newsletter and they're not a member.
- They're a member, but it's been a long time since they clicked a link in an email.
- Maybe someone **is** a subscriber, but they're reading us in their Facebook app -- the cookies between the app and their normal browser aren't shared, so we wouldn't know.
- Similarly, if someone gets our newsletter to their personal email, and today they're using their work computer.
- Or, someone forwarded a newsletter to a friend. When the friend clicks a link, we'll think they're the original recipient.
- A known user that is visiting in private or incognito mode, for their own reasons.
- A known user has cleared their browsing data and has not yet reconnected to their account.

There's no good way around these inaccuracies. So, when considering the numbers, don't sweat small digits -- look at the data in aggregate. If one story was read by three more members than another, that doesn't really mean anything.

Contact 30-day reading

When a user reads a post, we write it down in a cookie on their computer. And then on each page view, we count the number of posts visited in the last 30 days and send that to Google Analytics when we register the pageview.

Every user will have **one or more** (see note below) of these values for the *Contact 30-day reading* dimension.

0 posts	Users who have visited our site, but haven't read an article. For example, they just visited the home page.
1 post	Users who read one post in the last 30 days.
2-3 posts	Users who read 2-3 posts in the last 30 days.
4-5 posts	etc.
6-8 posts	etc.
9-13 posts	etc.
14-21 posts	etc.
22-34 posts	etc.
35-55 posts	etc.
56+	Users who read 56 or more posts in the last 30 days.

Note: Reading is tracked in a different way than the other dimensions. When you're looking at this data, know that the users tracked in higher groups have been counted in lower groups. For example, on the way to reading 4-5 posts, you would also have read 2-3 posts and 1 post.

Want to see exactly how this code all works? [Scroll down!](#)

How to use custom dimensions once you've got them all set up

Alright! The fun stuff! Here are a couple examples of how you might use the Google Analytics UI to look at our custom dimension data. We've also got some awesome automated reports to share. To check those out, [scroll down to the example reports section](#).

(If you don't yet have your own custom dimensions to play with, try using the user's age or location to get the general idea.)

What articles did members read most?

Go to Google Analytics -> Behavior -> All Pages. What you'll see is a list of the pages people viewed in the timeframe, sorted by the number of visits. Now we've got to filter the list. Add a secondary dimension to the list, like so:

The screenshot shows the Google Analytics 'All Pages' report. The primary dimension is 'Page'. A secondary dimension 'Contact Member Level' is selected from a dropdown menu. The search bar contains 'contact'. The table displays metrics for various pages, with the first row being the aggregate and subsequent rows filtered by 'Contact Member Level'.
















Primary Dimension: Page	Secondary dimension: Contact Member Level	Pageviews	Unique Pageviews	Avg. Time on Page	Entrances	Bounce Rate	% Exit	Page Value
	Aggregate	174,448 % of Total: 100.00% (174,448)	160,915 % of Total: 100.00% (160,915)	00:07:32 Avg for View: 00:07:32 (0.00%)	106,504 % of Total: 100.00% (106,504)	19.68% Avg for View: 19.68% (0.00%)	61.05% Avg for View: 61.05% (0.00%)	\$0.00 % of Total: 0.00% (\$0.00)
1. /	Member	21,523 (12.34%)	17,810 (11.07%)	00:02:43	16,952 (15.92%)	49.47%	50.16%	\$0.00 (0.00%)
2. /2019/01/06/hilltop-zoning-city-council/	Member	6,480 (3.71%)	6,105 (3.79%)	00:08:07	5,323 (5.00%)	10.33%	85.11%	\$0.00 (0.00%)
3. /2019/01/06/hilltop-zoning-city-council/	Member	3,948 (2.26%)	3,394 (2.11%)	00:08:24	2,906 (2.73%)	4.44%	81.10%	\$0.00 (0.00%)
4. /2019/01/06/hilltop-zoning-city-council/	Member	3,569 (2.05%)	3,348 (2.08%)	00:12:15	2,522 (2.37%)	12.39%	78.01%	\$0.00 (0.00%)

And then click the “advanced” link next to the search box in the screenshot above. Then, add a filter to only show metrics for dimension for users whose Member Level **begins with** “Member”, like so:

The screenshot shows the 'Advanced' filter configuration interface. It includes a filter type dropdown set to 'Include', a dimension dropdown set to 'Contact Member Level', a filter type dropdown set to 'Begins with', and a text input field containing 'Member'. Below this, there is a section for adding additional filters, currently empty.

We use **begins with** because members might be “Member 1”, “Member 2”, etc. and we want them all.

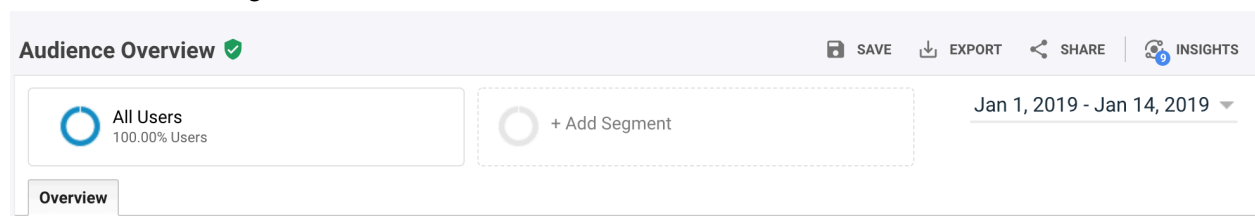
So! What do we see? Well, it looks like members visit our homepage most. And their most-read posts are about zoning and housing. Surprise! They're nerds!

Page 	Contact Member Level  	Pageviews 	Unique Pageviews 	Avg. Time on Page 	Entrances 	Bounce Rate 
		4,869 % of Total: 2.79% (174,448)	4,483 % of Total: 2.79% (160,915)	00:06:47 Avg for View: 00:07:32 (-9.91%)	2,232 % of Total: 2.10% (106,504)	12.68% Avg for View: 19.68% (-35.57%)
1. / 	Member 2	208 (4.27%)	164 (3.66%)	00:02:17	145 (6.50%)	35.21%
2. / 	Member 1	164 (3.37%)	129 (2.88%)	00:01:54	116 (5.20%)	38.05%
3. /2019/01/06/hilltop-zoning-city-council/ 	Member 1	40 (0.82%)	35 (0.78%)	00:10:45	19 (0.85%)	5.00%
4. /2019/01/07/whos-next-housing-14-rising-stars-in-the-denver-area/?mc_cid=2c8e396d4b&mc_eid=5d00c65b31 	Member 2	39 (0.80%)	28 (0.62%)	00:04:38	28 (1.25%)	6.90%
5. /2019/01/07/whos-next-housing-14-rising-stars-in-the-denver-area/ 	Member 2	34 (0.70%)	31 (0.69%)	00:09:12	14 (0.63%)	6.67%
6. /2019/01/07/whos-next-housing-14-rising-stars-in-the-denver-area/ 	Member 1	33 (0.68%)	27 (0.60%)	00:09:15	13 (0.58%)	14.29%
7. /2019/01/06/hilltop-zoning-city-council/ 	Member 2	32 (0.66%)	31 (0.69%)	00:10:14	12 (0.54%)	0.00%

That said, the report above is only for a couple weeks of data. If you want to really know what members like, you should probably look at a longer timeframe.

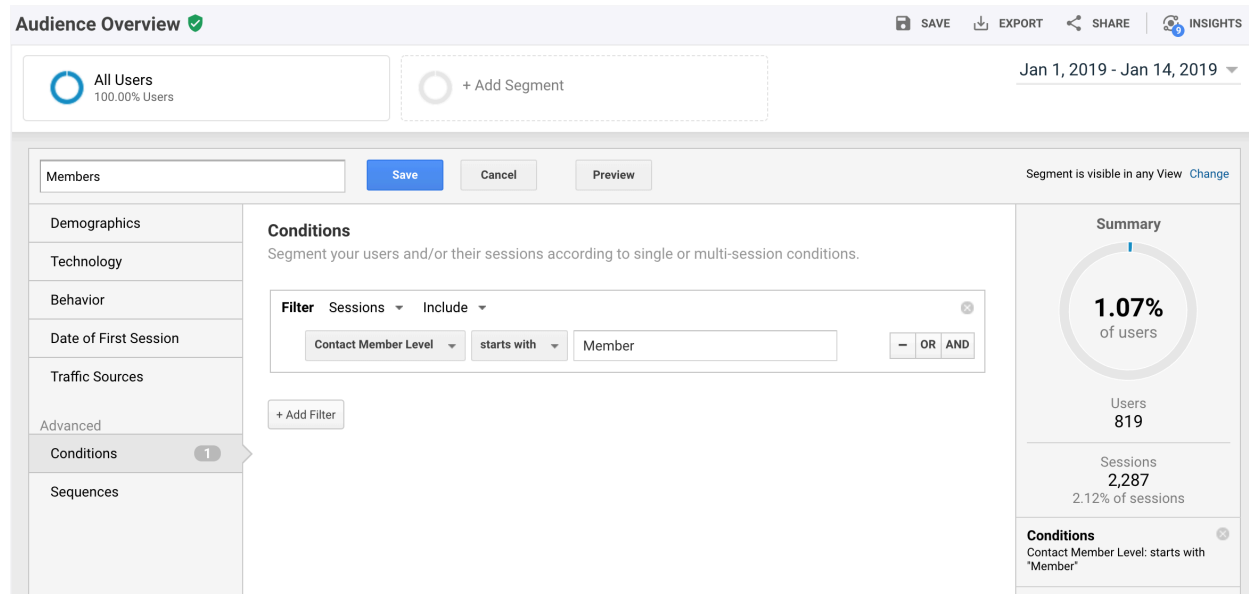
How do our different users behave, compared to each other?

To more quickly compare behavior, it's helpful to create segments. Atop most pages in GA, you will find an Add Segment button. Here it is on the Audience Overview.

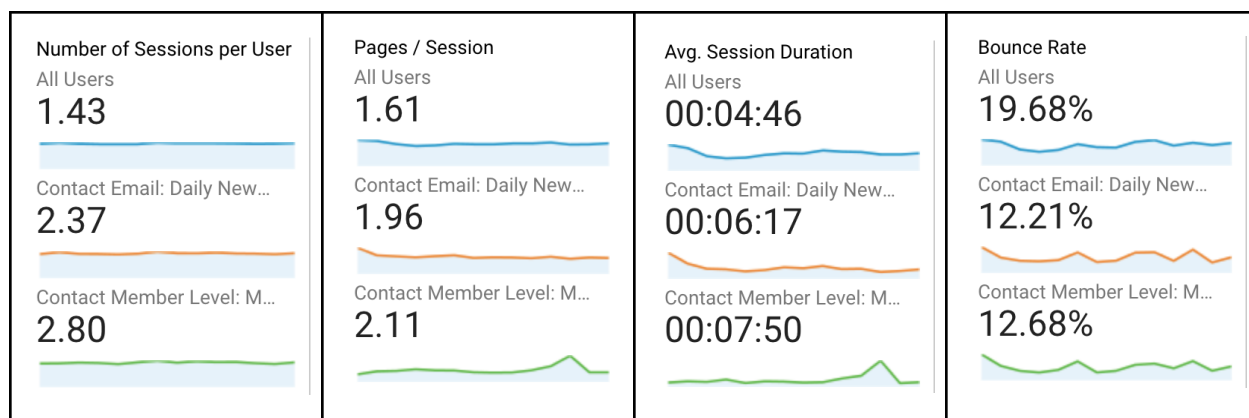


Let's add a few segments to our overview for comparison.

Click Add Segment, and then the big red NEW SEGMENT button. Give the segment a name, and under Conditions, specify a dimension and a value. Here we're looking at all Contact Member Levels that begin with "Member", which will match all levels of membership.



If we do that again for Contact Email / Daily Newsletter, we can compare our readers. Several of the stats we see here aren't particularly interesting. For example, it's no surprise that we have more newsletter subscribers than members reading our website. But the stats that are normalized, like sessions per user, are revealing:



In the charts above, the blue line represents all users, the orange line is people who get our daily newsletter, and the green line is members.

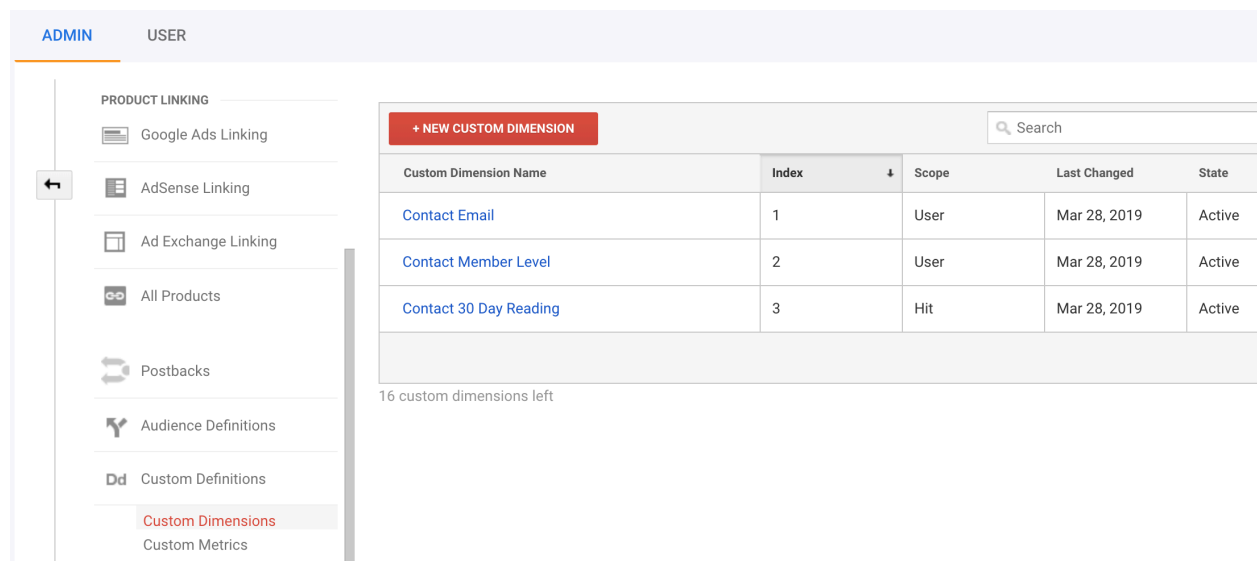
Newsletter readers and members have more sessions, more pages/session, a longer session duration, and a lower bounce rate! That's... not surprising! But good to know! And now it'd be interesting to look more closely at our content. [Check out this demo report](#) that breaks down the top stories on our site for all users, newsletter subscribers and members.

How we did it

Before we get to the code, let's look at how this is all configured in Google Analytics and see what custom dimensions look like in action. If you'd like to follow along at home, install the [Google Analytics Debugger add-on for Chrome](#).

Configuration in GA

In the admin settings for your Google Analytics property, you'll see a section for Custom Definitions, and under that, Custom Dimensions. We've got three dimensions configured, like so:



The screenshot shows the Google Analytics Admin interface. On the left is a sidebar with a menu under 'ADMIN' containing 'PRODUCT LINKING', 'AdSense Linking', 'Ad Exchange Linking', 'All Products', 'Postbacks', 'Audience Definitions', 'Custom Definitions', 'Custom Dimensions' (highlighted in red), and 'Custom Metrics'. The main content area is titled 'CUSTOM DIMENSIONS' and features a '+ NEW CUSTOM DIMENSION' button and a search bar. Below these is a table with the following data:

Custom Dimension Name	Index	Scope	Last Changed	State
Contact Email	1	User	Mar 28, 2019	Active
Contact Member Level	2	User	Mar 28, 2019	Active
Contact 30 Day Reading	3	Hit	Mar 28, 2019	Active

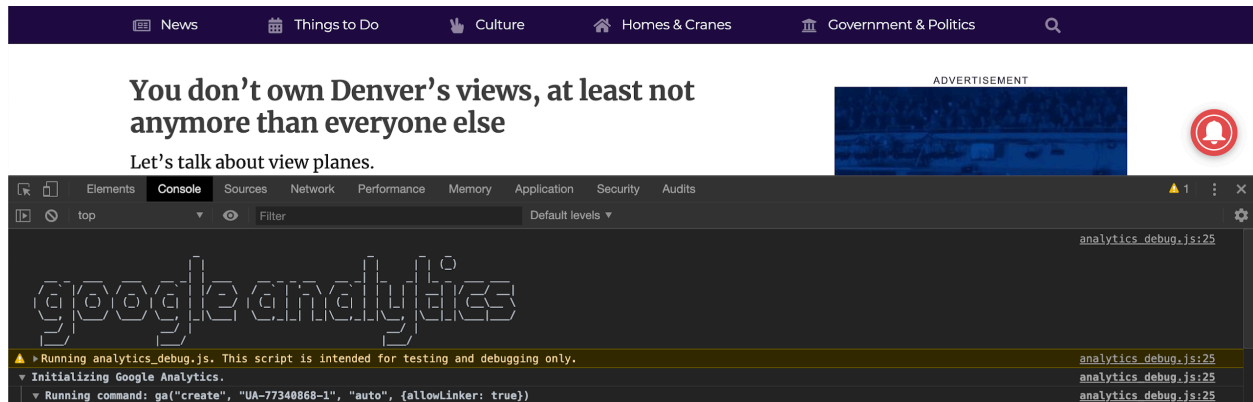
Below the table, it says '16 custom dimensions left'.

There's not much to configure, apart from the name of the dimension, and the scope. The scope of a metric may be tricky to get your head around at first... [you might want to read up](#). We use User-level scope for Member Level and Email because we want our knowledge of the user to persist across sessions. And we use Hit scope for 30-day reading because the number of articles you've read will change with each hit.

The values of the dimensions ("4-5 posts", "6-9 posts", "Daily Newsletter", etc.) are not configured here. We send the values to Google Analytics when the pageview is registered. Here's where the analytics debugger comes in handy.

What gets sent to GA?

Enable the Google Analytics Debugger and visit [a page on Denverite](#). Then open the Javascript Console. (View > Developer > Javascript Console) You'll see a whole bunch of stuff in the console about Google Analytics, like so:



If you scroll down into that, you'll see our custom dimensions in action:

clientId	(&cid)	721093619.1583344241
dimension1	(&cd1)	Daily Newsletter
dimension2	(&cd2)	None
dimension3	(&cd3)	4-5 posts

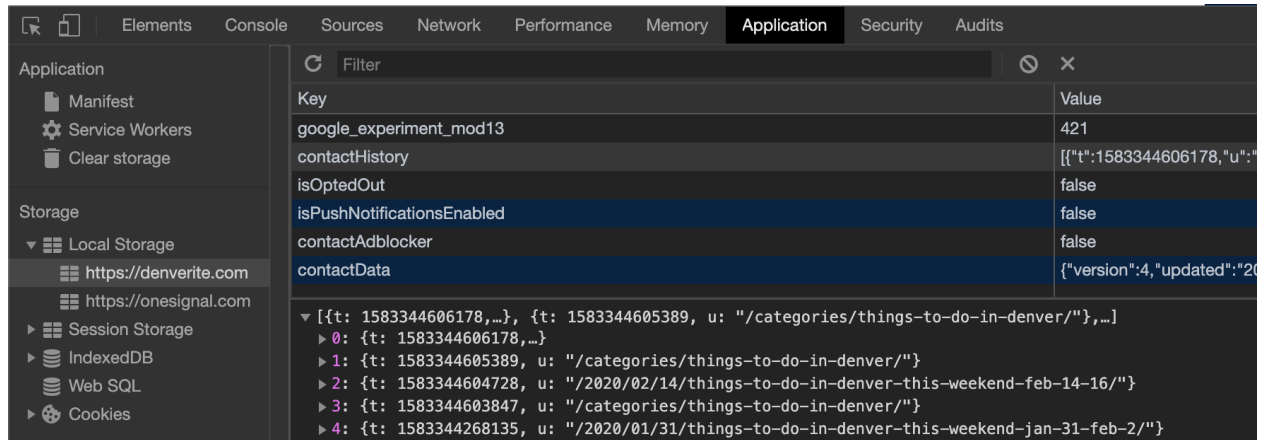
- dimension1 is "Contact Email", and since I receive the newsletter, it's set.
- dimension2 is "Contact Member Level", and I'm not a member of Denverite, so it's None.
- dimension3 is "Contact Reading Frequency", and I've read 4-5 posts on Denverite so far this month.

So far, this is pretty simple, but how the heck do we know this information about our readers? Let's go a little deeper.

Cookies!

The information that we send to GA is kept in a couple of cookies on the user's browser. (Technically, the data is in "Local Storage", not cookies, but that difference doesn't really matter for this conversation.)

Click the Application tab that's at the top of your Javascript Console. In the left-hand menu, click Local Storage > <https://denverite.com>. And then click contactHistory in the list. You'll see something like this:



That ugly stuff at the bottom is the list of articles our reader has clicked on in the last 30 days. The number next to "t:" is a timestamp. (1583344605389 = Wednesday, March 4, 2020 11:56:45.389 AM GMT-06:00) and next to "u:" is the path to the page on our site.

So! When the page loads, we have a little bit of code that reads that cookie, counts the articles in the contactHistory list, and sets a value that we pass to Google Analytics as dimension3. (We'll get to the actual code in a moment.)

The email and membership information is stored similarly, under contactData:

contactData	{"version":4,"updated"
<pre> ▼ {version: 4, updated: "2020-03-04T17:50:42+00:00",...} version: 4 updated: "2020-03-04T17:50:42+00:00" ▼ data: {mc_id: "b31feb6b33", since: "2017-03-14T12:04:54+00:00", subscribed_to_list: true,...} mc_id: "b31feb6b33" since: "2017-03-14T12:04:54+00:00" subscribed_to_list: true newsletter_subscriber: true breaking_news_subscriber: true rating: 5 current_member: false member_level: 0 recurring_member: false suggested_recurring_amount: 5 no_promote: false major_donor: false member_expiration: "2019-06-26" donate_7: false donate_14: false donate_30: false </pre>	

And, again, there's a bit of code that reads the cookie and decides what values to send for dimension1 and dimension2 -- Contact Email and Contact Member Level, respectively.

But how do the cookies get set?

So, we keep the data in a cookie. But how do the cookies get set?

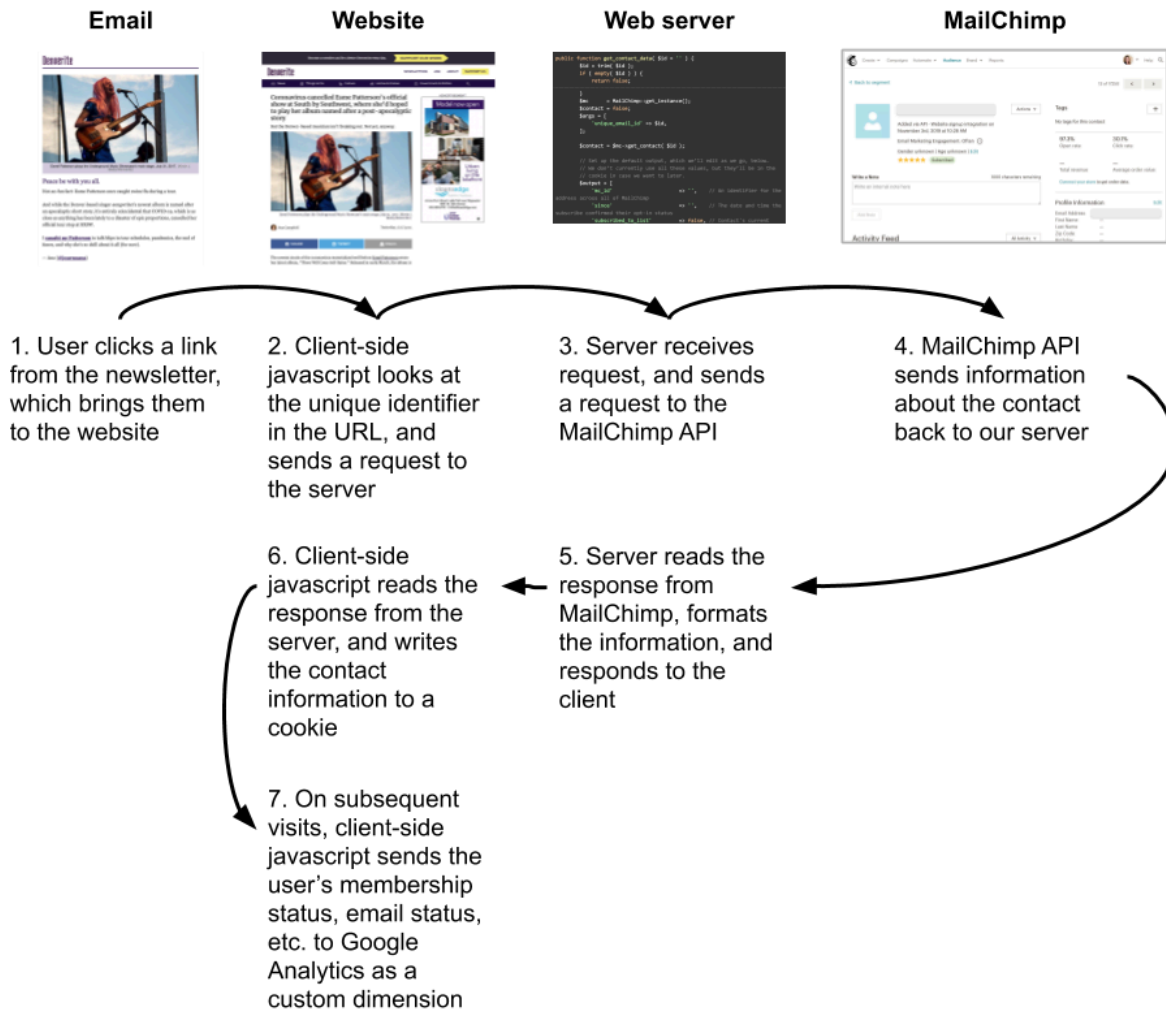
Reading frequency is relatively easy to track. On every page load, we update the cookie with the path and timestamp for the page the user loaded. See the example code later in the doc.

Membership and email, on the other hand, are a bit tricky.

I mentioned this earlier, but let's dig deeper. When someone clicks a link in one of our emails, there's a little code attached at the end:

https://denverite.com/2018/11/15/colorado-democrats-will-be-in-control-of-the-states-general-as-sembly-next-year-now-what/?mc_cid=f3d5611796&mc_eid=b31feb6b33. The code (in pink) can be used to identify the person to whom the email was sent.

When someone visits our website, and that identifier is on the URL, we look them up in MailChimp and write their email subscription and membership status to the cookie. And on subsequent visits, that information is sent to Google Analytics as our custom dimension values. There's also some example code for this process, later in the doc.



What about just asking folks to log in?

All these API calls might seem a bit convoluted. A much simpler solution would be to ask our readers to log in to our website. Then we'd just know who they are and it'd be trivial to send that information to GA. But we **really** don't like that solution. Users don't want to log in to your website. And I don't want to build a login system.

Yes, it would produce more accurate analytics for the people who log in, but what about the folks who don't log in? We don't have a paywall and we don't have a comments system, so we really don't have any good reason to ask people to log in to our sites. And we're good with that. :)

Alright! So that's how it all works, conceptually. What's the actual code look like?!

Example code (Yes! Finally!)

Sadly, we never built this system to be easily reused. It's not a fancy wordpress plugin you can just install and use. But if you've got a programmer handy, it's not particularly complicated to implement. Here are the important bits.

Writing the reading frequency data

On page load, this function gets called. It uses a helper called [localStorageCookie](#).

```
contactHistory() {  
  var history = localStorageCookie(this.historyStorageKey);  
  if (! history || ! Array.isArray(history)) {  
    history = [];  
  }  
  
  // Log the current timestamp and URL path  
  history.unshift({  
    t: Date.now(),  
    u: window.location.pathname  
  });  
  
  // Remove items that are too old  
  var maximumNumberOfDays = 30;  
  var dateCutoff = new Date();  
  dateCutoff.setDate(dateCutoff.getDate() - maximumNumberOfDays);  
  history = history.filter((item) => {  
    return (item.t > dateCutoff.getTime());  
  });  
  
  // Store the result  
  localStorageCookie(this.historyStorageKey, history);  
}
```

First it grabs the history (the historyStorageKey is "contactHistory"), then it writes a new entry to the history, removes any entries that are older than 30 days, and stores the new history.

Writing the email and membership data

A pair of client-side javascript functions, `contactData` and `fetchData`, do the work to retrieve and write the email and membership data, in conjunction with a web service running on our server, which we'll get to in a moment.

```
contactData() {

    var queryStringId = getURLParams('mc_eid');
    var contactData = localStorageCookie(this.dataStorageKey);

    var theId = contactData.data.mc_id;
    // Query string ID takes precedence over ID from cookie
    if (queryStringId) {
        theId = queryStringId;
    }

    // Check if the cookie version is out of date
    if (contactData.version != this.version) {
        this.deleteData();
        this.fetchData(theId);
        return;
    }

    // Check if the cookie is stale...
    var validNumberOfDays = 14;
    // Get now in seconds since epoch
    var now = new Date().getTime() / 1000;
    // Get our last updated time in seconds since epoch
    var updatedCutOff = new Date(contactData.updated).getTime() / 1000;
    // Add the amount of seconds to determine our cutoff timestamp
    updatedCutOff += 60 * 60 * 24 * validNumberOfDays;

    // If the cutoff date is in the past we need to refresh the data
    if (now >= updatedCutOff) {
        this.deleteData();
        this.fetchData(theId);
        return;
    }
}
```

contactData first grabs the mc_eid parameter from the querystring (that's the magic code that lets us look up a user in MailChimp). Then it does a handful of checks to see if the data in our cookie is still fresh, and fetches new data if it's not.

```
fetchData(id) {  
    // Don't bother to make an AJAX request if our id is false  
    if (!id) {  
        return;  
    }  
    var storageKey = this.dataStorageKey;  
    var ajaxData = {  
        action: 'get_contact_data',  
        contactID: id  
    };  
    $.post('URL FOR THE WEB SERVICE', ajaxData, (resp) => {  
        if (!resp.success) {  
            return;  
        }  
        localStorageCookie(storageKey, resp.data);  
    });  
}
```

fetchData makes an AJAX request to a web service running on our server. We must do this lookup server-side because we're accessing the MailChimp API, which requires an API key, and we don't want to expose our key to the world.

On the server side, the concepts are relatively straightforward -- using the mc_eid, look up the user via the MailChimp API, parse the API response, and put the data into a useful format. Unfortunately, our code is somewhat tangled up in all the other business of our website, so it's not a copy-and-paste job. Instead, I'll illustrate a few key concepts with code snippets.

When fetchData makes the AJAX request, we call this function, which accepts the mc_eid of the user.

```
public function get_contact_data( $id = '' ) {  
    $id = trim( $id );  
    if ( empty( $id ) ) {  
        return false;  
    }  
    $mc      = MailChimp::get_instance();  
    $contact = false;  
}
```

```

$args = [
    'unique_email_id' => $id,
];

$contact = $mc->get_contact( $id );

// Set up the default output, which we'll edit as we go, below.
// We don't currently use all these values, but they'll be in the
// cookie in case we want to later.
$output = [
    'mc_id' => '', // An identifier for the
address across all of MailChimp
    'since' => '', // The date and time the
subscribe confirmed their opt-in status
    'subscribed_to_list' => false, // Contact's current
status
    'else' => false, // Are they subscribed to the Daily
Newsletter group?
    'breaking_news_subscriber' => false, // Are they subscribed
to the Breaking News group?
    'rating' => 1, // Star rating for this
member, between 1 and 5
    'current_member' => false, // Status of contact's
membership
    'member_level' => 0, // Current level of
membership
    'recurring_member' => false, // Is contact a
recurring member?
    'suggested_recurring_amount' => 0, // Suggested amount for
recurring membership
    'no_promote' => false, // ???
    'major_donor' => false, // Has the contact
donated $500 or more?
    'member_expiration' => 0, // When does their
membership expire?
    'donate_7' => false, // Has contact donated
in the past 7 days?
    'donate_14' => false, // Has contact donated
in the past 14 days?
    'donate_30' => false, // Has contact donated
in the past 30 days?
    'donate_365' => false, // Has contact donated

```

in the past 365 days?

```
];
if ( ! $contact ) {
    return $output;
}

// If the response back from Mailchimp is a 404 then bail
if ( ! empty( $contact->status ) && '404' == $contact->status ) {
    return $output;
}

$output['mc_id'] = $contact->unique_email_id;
$output['since'] = $contact->timestamp_opt;
if ( 'subscribed' == $contact->status ) {
    $output['subscribed_to_list'] = true;
}
$output['rating'] = intval( $contact->member_rating );

$merge_fields                                = $contact->merge_fields;

// Then we look up the merge fields
// ONLY SHOWING ONE HERE, AS AN EXAMPLE
// YOUR MERGE FIELDS WILL BE DIFFERENT :)
$output['current_member']                    = $merge_fields['BPCURMEM'];
// etc, etc...

// Then we iterate over the groups the user is subscribed to
// and look for the two we care about, daily and breaking news.
$email_groups = Email_Groups::get_instance();
$newsletter_groups = $email_groups->get_groups( 'Newsletters' );
$newsletter_groups = $newsletter_groups->groups;
$the_groups = array_merge( $newsletter_groups, $story_groups );
foreach ( $contact->interests as $id => $val ) {
    if ( $val ) {
        foreach ( $the_groups as $group ) {
            if ( $id == $group->id ) {
                switch ( $group->name ) {
                    case 'Daily Newsletter':
                        $output['newsletter_subscriber'] = true;
                        break;

                    case 'Breaking News':
```

```

        $output['breaking_news_subscriber'] = true;
        break;
    }
}
}
}
}
return $output;
}

```

The middle part of the code above, where we grab the merge field values, will be different depending on your setup. If you're a News Revenue Hub client, all sorts of useful information will be synced to your merge fields, but you could also use a tool like Zapier to synchronize between your CRM and MailChimp.

I have not included the code we use to actually access the MailChimp API. Our MailChimp object is somewhat complicated because we used the API to do many, many things. For example, creating and deleting groups. It's messier than you need to read. And if you've gotten this far, you probably know how to write a bit of PHP that hits an API and returns the results.

Reading the cookie and sending the data to Google Analytics

Oh my! It's taken us a long time to get here. But here's the payoff. When we initialize our Google Analytics code, we read the what's stored in contactData and contactHistory, set the values for our three custom dimensions, and send the pageview.

```

ga('create', 'YOUR GOOGLE ANALYTICS ID', 'auto', { 'allowLinker': true });
ga('require', 'displayfeatures');

// Set up custom dimensions
ga((tracker) => {

    let contactEmailDimension = 'Unknown';
    let memberLevelDimension = 'Unknown';
    const contactData = localStorageCookie('contactData');
    if (contactData && 'data' in contactData) {
        const data = contactData.data;

        // Contact Email Dimension
        if (data.subscribed_to_list) {

```

```

    contactEmailDimension = 'Other';
  }
  if (data.newsletter_subscriber) {
    contactEmailDimension = 'Daily Newsletter';
  }

  // Contact Member Level Dimension
  if (!data.current_member && !data.donate_365) {
    memberLevelDimension = 'None';
  }
  if (!data.current_member && data.donate_365) {
    memberLevelDimension = 'Donor';
  }
  // In theory this conditional should never be triggered but if
something
  // goes wrong with data.member_level this will have us covered
  if (data.current_member) {
    memberLevelDimension = 'Error';
  }
  if (data.current_member && data.member_level > 0) {
    memberLevelDimension = 'Member ' + data.member_level;
  }
}
tracker.set('dimension1', contactEmailDimension);
tracker.set('dimension2', memberLevelDimension);

// Contact Post Visiting Frequency
let contactFrequency = '0 posts';
const contactHistory = localStorageCookie('contactHistory');
if (contactHistory) {
  // Only count posts
  const posts = contactHistory.filter((item) => {
    return (item.u.slice(0, 3) === '/20');
  });
  const postCount = posts.length;
  if (postCount == 1) {
    contactFrequency = '1 post';
  } else if (postCount >= 2 && postCount < 4) {
    contactFrequency = '2-3 posts';
  } else if (postCount >= 4 && postCount < 6) {
    contactFrequency = '4-5 posts';
  } else if (postCount >= 6 && postCount < 9) {

```

```

    contactFrequency = '6-8 posts';
  } else if (postCount >= 9 && postCount < 14) {
    contactFrequency = '9-13 posts';
  } else if (postCount >= 14 && postCount < 22) {
    contactFrequency = '14-21 posts';
  } else if (postCount >= 22 && postCount < 35) {
    contactFrequency = '22-34 posts';
  } else if (postCount >= 35 && postCount < 56) {
    contactFrequency = '35-55 posts';
  } else if (postCount >= 56) {
    contactFrequency = '56+';
  }
}
tracker.set('dimension3', contactFrequency);
});

ga('send', 'pageview');

```

Example reports!

We've built many reports that segment our audience by our custom dimensions. Here are a couple examples.

The reading habits for different user segments

Our [most viewed pages report](#) looks back at the last 90 days, and shows the top pages for members, newsletter subscribers, and all readers.

Everyone	Members	Newsletter subscribers
Denverite, the Denver site!	Denverite, the Denver site!	Denverite, the Denver site!
17-year-old killed in shooting at Aurora mall - Denverite, the Denver site!	Median property value rose 50 percent in 2 years in Elyria-Swansea and Globeville -- again - Denverite, the Denver site!	The public owns Denver's Union Station but now only people with money can lounge there - Denverite, the Denver site!
You're damn right this Park Hill 'tuberculosis house' is historic. A KKK adversary and a mountaineering giant lived there. - Denverite, the Denver site!	What smells where in Denver - Denverite, the Denver site!	Median property value rose 50 percent in 2 years in Elyria-Swansea and Globeville -- again - Denverite, the Denver site!
You're not allowed to tip Denver bus drivers but you are allowed to tip Denver bus drivers - Denverite, the Denver site!	The public owns Denver's Union Station but now only people with money can lounge there - Denverite, the Denver site!	"Practically instant housing": This was a hotel. Now it's a place for people who've been trying to find a place to live in Denver. - Denverite, the Denver site!
What smells where in Denver - Denverite, the Denver site!	Do you know Denver's rules about living together? Here's what some people want to change about them. - Denverite, the Denver site!	What smells where in Denver - Denverite, the Denver site!
The public owns Denver's Union Station but now only people with money can lounge there - Denverite, the Denver site!	Tom's Diner added to National Register of Historic Places, preserving its life on Colfax - Denverite, the Denver site!	Tom's Diner added to National Register of Historic Places, preserving its life on Colfax - Denverite, the Denver site!

Read the “How it works” section of our [Measuring local users documentation](#) to understand how we’re pulling the data from Google Analytics. In addition, the articles are each assigned a unique color, based on their title, which makes it easier to spot the same article in different columns. Go to Tools > Script editor if you’d like to see how that’s done.

FWIW, this report is somewhat interesting, but it would be even more interesting if we had an additional custom dimension to track story types. We never built this, but it would be relatively trivial to add an additional dimension that tracked whether the story was news, sports, entertainment, etc. Other news organizations also have a dimension for story authors. With either of those, you could then show which authors are most read by newsletter subscribers, which category of stories are most read by members, etc, etc.

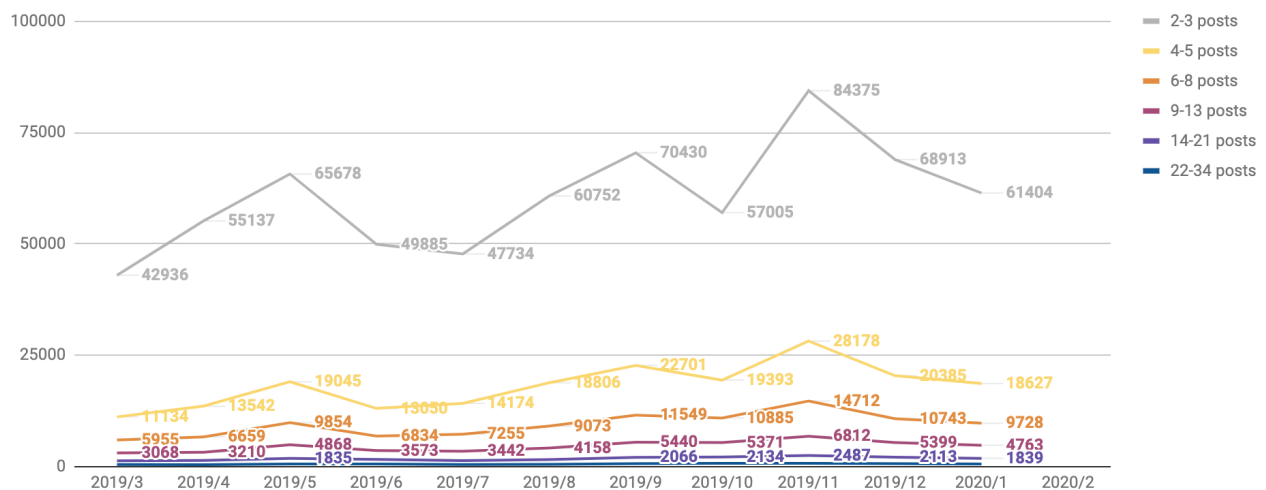
The number of people who read a lot of articles

One of the most important data points we track is the number of people who read more than five articles per month. We’ve been told by many smart people that this is your ceiling for membership -- if you want more members, you need more people who read at least five articles.

Our [30-day reading report](#) uses the custom dimension for reading frequency to show our progress towards creating a membership-ready audience.

How many people read at least X posts/ in the last 30 days?

Values are cumulative. i.e. all 4-5 post readers are also counted as 2-3 post readers.



This report also works similarly to our [Measuring local users](#) report, so check out those docs if you wanna fiddle with it. This one’s a bit more complicated, but the principles are the same.

22-34 posts	35-55 posts	56+
Billy Penn: Philly's source for local news, info and things to do	Billy Penn: Philly's source for local news, info and things to do	Meet Hannah Callowhill Penn, PA's first and only woman leader - On top of Philly news
Map: Philly coronavirus cases by ZIP code - On top of Philly news	Meet Hannah Callowhill Penn, PA's first and only woman leader - On top of Philly news	Billy Penn: Philly's source for local news, info and things to do
Meet Hannah Callowhill Penn, PA's first and only woman leader - On top of Philly news	Map: Philly coronavirus cases by ZIP code - On top of Philly news	Map: Philly coronavirus cases by ZIP code - On top of Philly news

We also have built a report to show the [top pages for these groups of readers](#). The results are somewhat stunning. The post in green came out of nowhere to be the most-read among extremely frequent readers!

What else can you do with this data?

All this work isn't just for reporting. You can also use the same tooling to drive conversions based on what you know about the user! [Check out the accompanying writeup!](#)

Who are you? What's Memberkit?

I'm [Brian Boyer](#). I used to be the VP of Product and People at Spirited Media. We were the home to [The Incline](#), [Denverite](#) and [Billy Penn](#). In 2019, Spirited was broken up. But -- and this is where it gets weird -- we had just completed [an accelerator program](#) and had received a grant to help our membership programs. So, I teamed up with Ashley Dean and Dave Burdick of Denverite, and Danya Henninger of Billy Penn to just do the project anyway.

This document and accompanying report are [one of many outcomes of our project](#), and we wanted to share what we learned.

The designs and code in this document were originally created by the Spirited Media product team: Russell Heimlich, Chris Montgomery, Livia Labate and myself. Huge thanks to Danya Henninger and Dave Burdick, my colleagues at Billy Penn and Denverite, respectively, for making it all available to share with the world.

Finally, Memberkit is a snazzy name I thought of a few months ago. Seemed like a good name for a membership toolkit! It's not a company, or software, or anything organized.

[Memberkit 1.0](#)

Copyright 2020, WHYY and Colorado Public Radio. This work is licensed under a [Creative Commons Attribution-NonCommercial 4.0 International License](#).