Getting Started with PowerTrack

Streaming, filtered Twitter Social Data

Prepared by the
Developer Platform Relations Team
@Gnip / @TwitterBoulder

Table of Contents

Introduction

A founding belief at Gnip is that Social data has unlimited value and near limitless applications.

More than 500 million Tweets are generated every day. There are tens of thousands of different conversations happening in parallel covering news topics, brands, locations, and every subject imaginable.

Bridging the divide between the continuously flowing mountain of data and the applications that can analyze this data is PowerTrack. The singular goal of PowerTrack is to allow users to define and refine rules so that only useful messages are delivered, and the signal-to-noise ratio (desired to undesired message ratio) is optimized.

This document looks at the basic concepts of PowerTrack.

First the mechanics of connectivity are covered, and what expectations are placed on the connecting application.

Next a review of the message (called Activities) is done - describing what data and metadata is found in each message, and how elements are accessed.

Lastly, rules - the language of filtering - are covered, with examples and best practices for efficient rule management on the client side.

What is not covered in this document:
Out of scope for this document is how to store, process or analyze the data once received on the client side. The variety of use cases is vast, to the point where there is no singular best-practice.
There are some articles covering analysis available at blog.gnip.com for those wanting to understand how customers are analyzing Twitter data.

From Tweet to Activity to Customer

When a Tweet is created by a user or application, the message itself is at most 140 characters. It is sent to Twitter's service along with the user authentication information, and optionally geo tagging information. Internally, a record is made of the users's profile at that moment, the date/time of the Tweet, hashtags and urls that were referenced in the message, and a unique ID that was generated for it. If it was a ReTweet, additional information is stored about the original Tweet that was referenced.

The Tweet in original format (Twitter native JSON) is then sent to Gnip for additional processing via a stream called the Firehose. ***Note: Certain accounts are blocked and/or marked as spam by Twitter and while those accounts may be viewed directly on Twitter, messages from those accounts will not be sent over the Firehose.***

Gnip receives the message and performs three significant actions: First the original format is transformed into Activity Format - an expanded JSON structure that allows for more data elements. Next, enrichments are performed to add additional metadata to the message. These include profile geo-tagging, URL unwinding and others. Finally, the data is sent to multiple systems for storage and realtime delivery. The messages are sent to a queue that will feed the Activity to customers that have rules in place matching on it's data or metadata. The Activity format record averages ~ 1500 bytes per record and can be well over 6,000 bytes long. The time to process the message, from creation to delivery averages around 15 seconds

PowerTrack Features

Only the Necessary Data
PowerTrack is Gnip's enterprise-grade filtering language that delivers the ability to get complete coverage of the data necessary. Filtering capabilities exist for geo-boundaries, keyword and phrase matches, the presence of links or images, the language of an activity and much more.
A wide range of operators can be used to easily filter for the exact data required.

Format Normalization
The Activity Streams format from Gnip enables the consumption of all of data in a single standardized format across all available data sources.

URL Expansion
Gnip expands short URLs (such as links from t.co, bit.ly, and many others) and provides the extended URLs as added metadata to activities.

Language Detection
Gnip uses a language detection algorithm to apply language metadata that is included in the activity. This premium feature currently support 24 languages.

Basic Klout
Klout is the standard for online influence. The Basic Klout enrichment from Gnip appends Klout Scores to activities from the Twitter stream.

Premium Klout
A premium feature, Klout Topics can be added, delivering with it enterprise licensing terms necessary to store Klout data longer than 7 days and incorporate social influence into applications.

Profile Geo
Gnip's Profile Geo enrichment significantly increases the amount of usable geodata for Twitter, it normalizes unstructured location data to common geographies and provides latitude/longitude coordinates for those places.

Rules API
Filtering of data is done via rules. Rules are managed via a simple API, and changes can be submitted real-time with no stream disconnect/reconnects required.

Dude Where's My Tweet?
Gnip keeps a record of all the data to each customer. If there's a question about the delivery of an activity support will track it down. The goal is to provide customers with every single activity necessary.

Backfill (optional)
Backfill automatically protects against data loss caused by brief disconnects. Activities are queued up, and if the stream is reconnected within five minutes, data delivery picks up right where it left off.

Replay
A premium feature, Replay provides access to data that was missed between five minutes and five days. All standard stream rules are respected, and the data is delivered via a secondary connection so the live connection is not affected.

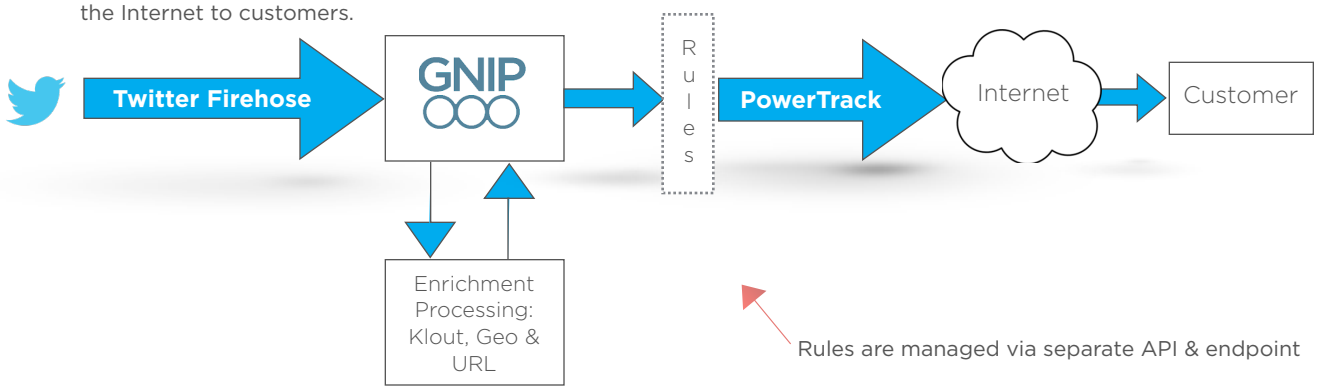
Redundant Stream
A premium feature, redundant streams can mitigate the effects of a disconnect by providing a second live connection. The redundant stream can help bridge periods of disconnection and prevent potentially missed data.

Dedicated Support Team
Technical on-boarding, documentation, programming guidance and libraries on GitHub, and rapid response during business hours with 24/7 emergency support ensures success.

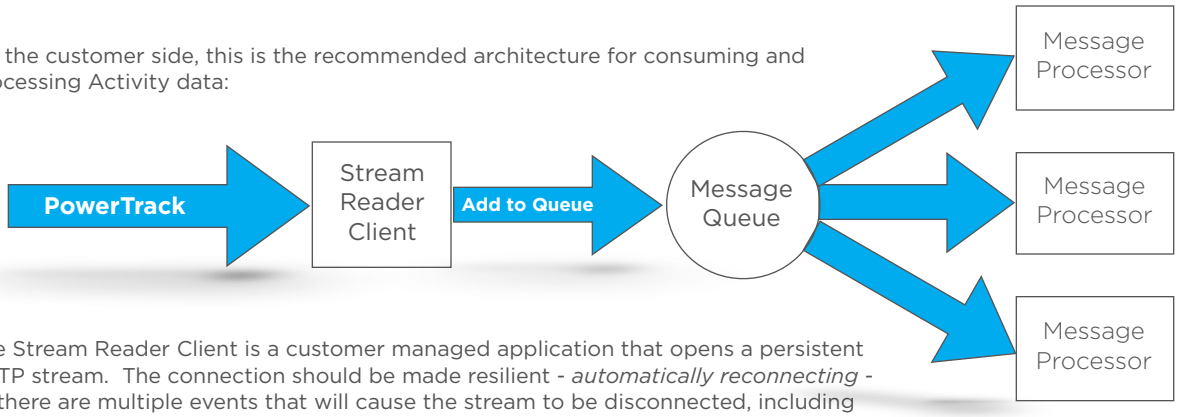
Architecture - How Activities are delivered

The diagram below shows (simplified) how Tweets flow from Twitter to end customers via a PowerTrack stream. Native format activities come in through the Twitter firehose as native-format JSON objects and are enriched with additional metadata. Some enrichments are standard while others are optional and incur an additional cost,

Rules are used to filter data to match customer specific needs. The streams of filtered activities are delivered over the Internet to customers.



On the customer side, this is the recommended architecture for consuming and processing Activity data:

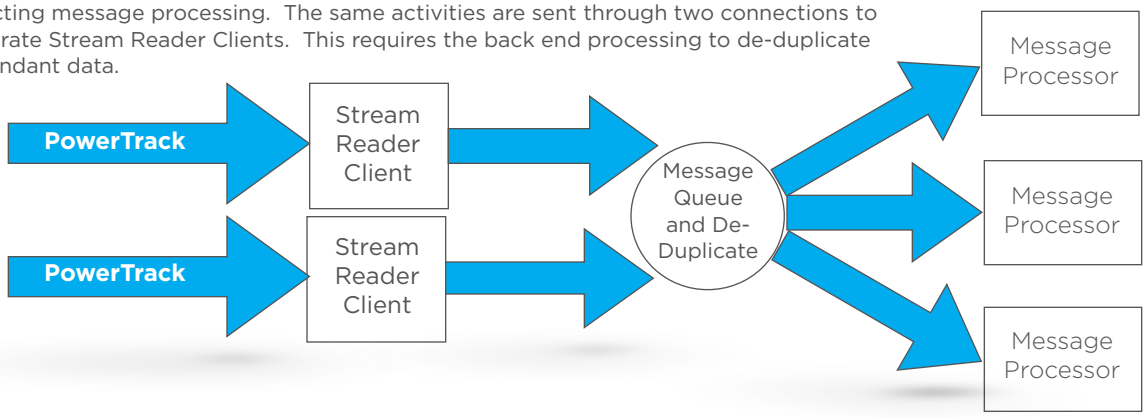


The Stream Reader Client is a customer managed application that opens a persistent HTTP stream. The connection should be made resilient - *automatically reconnecting* - as there are multiple events that will cause the stream to be disconnected, including regularly scheduled maintenance and transient Internet routing issues.

Activities are streamed in GZIPed block format to reduce bandwidth. Activities may be spread across blocks, so the Stream Reader Client software should tolerate incomplete activities within blocks. Keep-Alive messages - no activity data, just a carriage return - are sent every 15 seconds to indicate the stream connection is functioning.

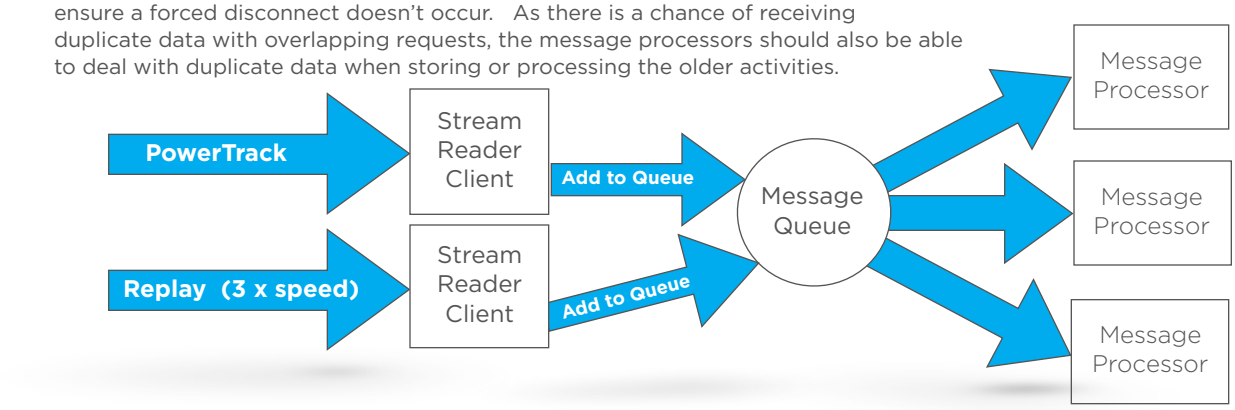
The Stream Reader Client should do *minimal processing* other than determining if an activity is completely downloaded, and add the complete activity JSON object into an external queue for processing. A separate message processor application should be used to perform any work required. The number of message processors should scale automatically based on message queue depth and message process time requirements.

The diagram below shows how a redundant connection can be used to prevent a failure of a stream reader from affecting message processing. The same activities are sent through two connections to separate Stream Reader Clients. This requires the back end processing to de-duplicate redundant data.



Replay

When a disruption in stream reading occurs for a period of time, within five days a Replay stream can be connected to in order to retrieve the missed activities. The Replay stream will operate at about 3x the speed of the normal PowerTrack stream, so handling that level of data (while also supporting the existing productions stream) should be considered when designing the architecture. When requesting a Replay stream via the API, a start & stop date/time is used. A best practice is to make small windows (like 4 to 6 hours per request) in order to moderate the stream and ensure a forced disconnect doesn't occur. As there is a chance of receiving duplicate data with overlapping requests, the message processors should also be able to deal with duplicate data when storing or processing the older activities.



Stream Reader Software

In any current software language, the process of opening and maintaining an HTTP stream connection is fairly straightforward. When combined with GZip compression, block management and best practices of Keep-Alive monitoring and connection management behavior, it can become a bit more complex. Fortunately, there are examples in several languages available on the support web site at http://support.gnip.com/code/powertrack_firehose_stream.html

Steam Reader software is expected to perform the following basic operations:

- ✓ Connect using stored credentials
- ✓ Detect a disconnect caused or indicated by:
  - Network disruption
  - Application failure
  - Forced disconnect due to Gnip software upgrade or other unexpected issues
  - Forced disconnect due to buffer full
  - Lack of HeartBeat
- ✓ Read blocks of data from HTTP stream - size of block should be variable (application setting)
- ✓ Decompress blocks via GZip decompression library
- ✓ Detect HeartBeat messages within stream
- ✓ Detect complete JSON activities, which may span across blocks
- ✓ Hand off complete JSON activities for processing by other applications

Message Processors

This is an application that takes complete activities and processes them in a way that is most appropriate for the business use case. It may involve storing the full activity to a NoSQL store, a SQL database or other storage. Ideally, it is designed so that multiple instances can be ran simultaneously to handle increases in load.

For situations where multiple client side applications are fed from a common PowerTrack connection/stream, the message processor can also act as an activity router - looking at the gnip.matching_rules array to determine which application(s) should be sent the activity.

Understanding Activities

Sample Activity JSON

Activities are delivered as a stream of JSON (JavaScript Object Notation) objects. Here is an excerpt (about 1/3) of a sample Activity object with the actual Tweet body highlighted. It contains information on the sender (actor), how the message was sent (generator), where it was sent from (geo/location) and much more.


```
{
    "id": "tag:search.twitter.com,2005:403224522679009280",
    "objectType": "activity",
    "actor": {
    "objectType": "person",
    "id": "id:twitter.com:1855784545",
    "link": "http://www.twitter.com/LandenEhlers",
    "displayName": "Landen Ehlers",
    "postedTime": "2013-09-11T23:52:03.000Z",
    "image": "https://pbs.twimg.com/profile_images/378800000646150423/83090ccb95a60def923c674e7bd002a0_normal.jpeg",
    "summary": "Senior Construction Science student at Texas A&M University. Barefoot Waterskiing National Champion.
                Vice President of the Texas A&M Waterski Team.",
          "friendsCount": 65,
          "followersCount": 15,
          "listedCount": 1,
          "statusesCount": 24,
          "twitterTimeZone": null,
          "verified": false,
          "utcOffset": null,
          "preferredUsername": "LandenEhlers",
          "languages": [
                          "en"
                 ],
          "location": {
                          "objectType": "place",
                          "displayName": "College Station, TX"
          },
          "favoritesCount": 2
    },
    "verb": "post",
    "postedTime": "2013-11-20T18:13:12.000Z",
    "generator": {
          "displayName": "Twitter for Android",
          "link": "http://twitter.com/download/android"
    },
    "provider": {
          "objectType": "service",
          "displayName": "Twitter",
          "link": "http://www.twitter.com"
    },
    "link": "http://twitter.com/LandenEhlers/statuses/403224522679009280",
"body": "Enjoyed our half price chicken and wawfuls today! @tamusportclubs @SullysGrill
         @TAMUWaterski #SCPartnerday http://t.co/XRsVqYy9Zo",
    "object": {
          "objectType": "note",
          "id": "object:search.twitter.com,2005:403224522679009280",
          "summary": "Enjoyed our half price chicken and wawfuls today! @tamusportclubs
            @SullysGrill @TAMUWaterski #SCPartnerday http://t.co/XRsVqYy9Zo",
          "link": "http://twitter.com/LandenEhlers/statuses/403224522679009280",
          "postedTime": "2013-11-20T18:13:12.000Z"
    },
    "favoritesCount": 0,
    "location": {
          "objectType": "place",
          "displayName": "College Station, TX",
          "name": "College Station",
          "country_code": "United States",
          "twitter_country_code": "US",
          "link": "https://api.twitter.com/1.1/geo/id/85128f80a57c03ad.json"
    }}
```

Filtering - Use and Construction of Rules

PowerTrack rules determine which activities are sent through the HTTP stream.

Each rule is independent of all other rules - there is no cascading or boolean logic between rules. Any activity that passes the logic for any rule will be sent over the connection. The number of rules can easily number in the tens of thousands. A separate system on the customer side should be used to manage rules, with processes used to synchronize them with the PowerTrack stream. A best practice is to keep track of the date/time rules were added/edited/deleted for internal auditing purposes.

Rule changes are made to a separate API endpoint than the PowerTrack stream itself, so rule changes do not disrupt an active stream. Changes to PowerTrack rules will affect the stream within 30 seconds in most cases.

Rule length

A named stream can either be configured for "Standard" or "Long" rules. Standard rules have a limit of 1024 characters and 30 positive and 50 negation operators or clauses. Long rules can be 2048 characters in length and do not have a positive or negative operator limitation. Because of the expanded capabilities, there is an additional charge for large rules. Standard rules are returned in the Activity payload in the activity.gnip.matching_rules field. For large rules, only the tag is returned.

	Positive Clauses	Negative Clauses	Total Rule Characters	Matching Rule **Value** Returned in Payload	Matching Rule **Tag** Returned in Payload
Standard Rule	30	50	1024	Yes	Yes
Long Rule	Unlimited	Unlimited	2048	No	Yes

Rule Tags

Each rule may be created with a tag. These tags have no effect on filtering, but can be used to create logical groupings or identification of rules within your app.
Here are some important things to know about rule tags:

- Each rule may have only one tag, with a maximum of 255 characters.
- Tags must be included with the rule as the rule is added.
- Multiple rules can use the same tag
- The tag is NOT a unique identifier by the Gnip API, but can be used as such way for internal systems,

JSON Formating

PowerTrack stream rules are managed for POSTing Add or Delete requests. Each named stream will have a unique rules API endpoint.

The format for the rules is a JSON object submitted as the "body" of the HTTP Post shown in the following format:

```
Content-type: "application/json"
{
    "rules":
    [
        {"value":"rule1","tag":"tag1"},
        {"value":"rule2"}
    ]
}
```

Note the "rules" array. and each rule submitted is separated by a comma. Notice the optional "Tag" value as well.

Rules with Double-quote String Literals and special characters

If the rule contains double-quote characters (") associated with exact-match or other operators, they must be escaped using a backslash to distinguish them from the structure of the JSON format. For example, if your rule is:

"social data" @gnip
The JSON formatted rule would be:
{"value":"\"social data\" @gnip"}

To include a double-quote character as a string literal within an exact-match, it must be double-escaped. For example, for a rule matching on the exact phrase 'Toys "R" Us', including the double-quotes around R, the plain-text representation of this would look like the following:
"Toys \"R\" Us"

Translating this to JSON format, you should use the following structure:
{"value":"\"Toys \\\"R\\\" Us\""}

Boolean Logic

There are five boolean operators for the rules.

The first is a white space " ". It is used as an implicit AND. For example "cat dog" as a rule (without quotes) can be read "return activities with both cat and dog tokens. ***Do NOT use the keyword AND unless ONLY messages with the word AND in them are desired.*** This is a very common mistake and will result in far few results than desired if used erroneously.

To change the above rule to be "return activities with either cat or dog ***or both*** tokens, an **uppercase** OR would be used, thus the altered rule would be "cat OR dog" Note that OR is not an Exclusive OR. Both sides matching will return a match.

"(" and ")" operators are used to group logic together. For example, "(cat OR dog) picture" would read "return activities of cats or dogs that also have picture". "cat OR (dog picture)" would read "return activities with cat, or dog that also have a picture.

Lastly, the negative sign '-' is used to negate or reverse the intent of another operator. "cat -dog" would translate as "return messages with cat but not dog in them."

Negation can be used with the parentheses to negate groups of operators. An example would be:
cat -(kitten OR lol OR cute) translated as "return messages with cat but not kitten or lol or cute in them. It could also be written as cat -kitten -lol -cute. This is not the same as "cat -kitten OR -lol OR -cute" ***Note: Using OR with a negated operator or keyword can cause a very high rate of data matches.***

Case Insensitivity

All operators evaluate data in a case-insensitive manner. For example, the rule Cat will match all of the following: cat, CAT, Cat.

Accents

In PowerTrack Operators, characters with accents or diacritics are treated the same as normal, unique characters and are not treated as word boundaries. For example, a rule of cumpleaños would only match activities containing the word cumpleaños and would not match activities containing cumplea, cumplean, or cumpleanos.

Tokenization

Each activity undergoes a process called tokenization as it is received by Gnip. Tokenization looks at the data in specific fields of the source JSON

Tokens are all case-insensitive, and delimited on punctuation including white space (spaces). This means that "coca-cola" is tokenized into three terms: coca, cola and coca-cola.

Languages that do not use white space (such as Japanese) do not tokenize well using this method. Instead, Japanese tokenization uses a grammatical model of Japanese plus a statistical model based on a word dictionary.
Because it's using a statistical model, the way it breaks up particular words may change depending on the context they are in. This method of tokenization breaks up parts of sentences reliably due to the grammar model, but it can have problems with compound words that are not in its dictionary. Katakana words are less likely to be broken up correctly, as they tend to be loan words.

Generally speaking, when you have changes in character sets, it appears that those are usually treated as token boundaries.

Note that rules (operators and keywords) are not tokenized, only activity data is.

The following fields of the JSON are tokenized and searched on for keywords:
activity.body
actvity.twitter_entity.hashtags
actvity.twitter_entity.urls
actvity.twitter_entity.user_mentions

PowerTrack Operators Overview

The following operators are available to filter the Twitter firehose via PowerTrack. These operators will match specific types of Tweets, and can be combined using standard PowerTrack syntax. Each operator is discussed individually with samples later in this document.

Text	People	Location	Language	Links	Attributes
Keyword has:hashtag #Hashtag “text match” contains “keywords near”~N	@mention from:user to:user retweets_of:user bio_contains bio_name_contains klout_score klout_topic klout_topic_id statuses_count followers_count friends_count listed_count is:verified has:mentions	point_radius bounding_box bio_location bio_location_contains time_zone place place_contains country_code has:geo has:profile_geo has:profile_geo_locality has:profile_geo_subregion profile_bounding _box profile_country_code profile_region profile_region_contains profile_subregion profile_subregion_contains	has:lang bio_lang twitter_lang lang	has:links url_contains	source sample generator has_media is_retweet retweets_of_status_id in_reply_to_status_id

Rule construction

It’s important to understand that every rule is a unique entity. Any Tweet matching any rule will be sent via the stream. There is currently no mechanism to create a rule-to-rule dependency.

In construction of rules sets, it is common to build a large number of rules. PowerTrack was designed to support this. Having consistency in how the rules are constructed will help with scale and overall rule management.

Good rule design starts with a pattern such as this:

“What am I looking for” (positive section)
“What am I filtering out” (negative section)
“tag” (how I want to identify data that passes this rule)

For example, if the intent is to find messages for “cats OR dogs OR rabbits” but not “puppies or kittens or bunnies”:

First start by creating a standard “filter rule phrase”

-(puppy OR puppies OR kitten OR kittens OR bunny OR bunnies)

Next create multiple rules - one for each positive phrase, using the same "filter rule phrase"

(cat OR cats) -(puppy OR puppies OR kitten OR kittens OR bunny OR bunnies)
(dog OR doggie OR dogs) -(puppy OR puppies OR kitten OR kittens OR bunny OR bunnies)
(rabbit or rabbits) -(puppy OR puppies OR kitten OR kittens OR bunny OR bunnies)

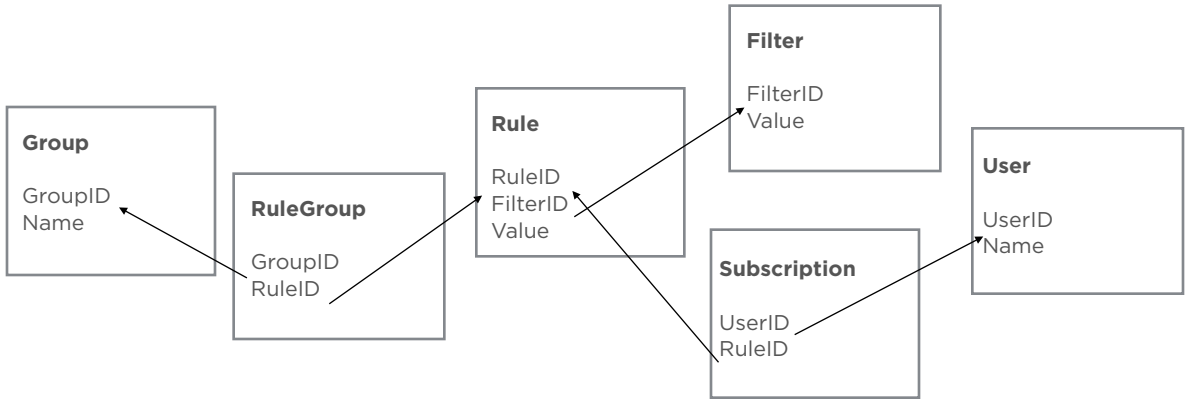
This can result in a lot of rules obviously, so tags allow us to group or identify them. For each of the rules above, it may make sense to use the same tag name “adultpets” as it doesn’t matter which individual rule it passed as the program will only care if it was cat, dog, or rabbit. Alternatively, a unique ID could be used and some external logic could group together the unique IDs as “adult pets”.

Rules management

While there's no one way to manage rules, a common technique is to create an external database. It can be as simple as a flat table of rules and tags, or more complex to handle large rule count complexities. Some best practices:

- Each rule is created with a unique identifier (such as a GUID) and as described above, a single search item.
- Additional information like a pointer to a common "filter clause" is stored.
- A separate process reviews changes to the database and generates the JSON and rules API management requests.
- As new rules are required by customers or other use cases, the existing rules can be checked to see if a fitting rule already exists and the new request "subscribed" to an existing rule.
- This method minimizes rule API calls and allows for rule usage to be tracked internally.
- This method supports the use case of multiple internal needs requiring the same rule getting tracked correctly.

Here's a sample database design for a rule management system. It supports having groups of related rules (Group), common filters (Filters) applied to multiple rules, and subscription to rules by users (could be "customers", or other appropriate pointer).



Activity Object Map

To understand the Activity object better, on the next page is an object map visualization of the JSON structure, followed by a data dictionary of the fields. The Activity object is the top level, with other objects like Actor, Generator and Provider nested inside of it. Some objects are arrays, for example the the "Topic" object is a nested array inside of klout_profile, which in turn is a nested object inside the Gnip object, which is nested inside of Activity. Bold and Italicized members refer to nested objects, thus activity.gnip.klout_profile.topic[] would refer to the array of topics associated with the actor.

Twitter Activity Record
JSON Schema Visualization

Data Dictionary

Activity Object

Property	Value	Description	Filtering operator(s)
id	String	A unique IRI for the Tweet. When storing Tweets, this should be used as the unique identifier or primary key.	
actor	Object	This object contains information about the entity that performed the activity.	
verb	String	Identifies the action that the activity describes. Valid options are ***post, share, delete, user_delete, user_undelete, scrub_geo, user_protect, user_unprotect, user_suspend, user_unsuspend, user_withheld*** and ***status_withheld.*** Note: Quoted Tweets are "post"s and are not the same Retweets.	is_retweet
generator	Object	This object describes the client application that generated the activity.	
provider	Object	This object describes the service the activity was posted through.	
inReplyTo	Object	This object contains a single field ("link") with a link to the original Tweet - only found in Retweet activities,	in_reply_to_status_id
location	Object	This object describes the "Place" where the Tweet was created. This is an object passed through from the Twitter platform.	
geo	Object	This object contains a point location where the Tweet was created. Note that most activities do not have this object.	has:geo
twitter_ entities	Object	The entities object from Twitter's data format. Note that in Retweets, Twitter may truncate the values of entities that it extracts at the root level. So, for Retweets, use object.twitter_entities to ensure that you are using non-truncated values.	
twitter_ extended_ entities	Object	An object from Twitter's native data format containing "media". This will be present for any Tweet where the twitter_entities object has data present in the "media" field, and will include multiple photos where present in the post. Note that this is the correct location to retrieve media information for multi-photo posts. Multiple photos are represented by comma-separated objects within the "media" array	
link	String	A Permalink for the Tweet in URL form.	
body	String	The Tweet text. In Retweets, note that Twitter modifies the value of the body at the root level by adding "RT @username" at the beginning, and by truncating the original text and adding an ellipsis at the end. Thus, for Retweets, object.body will contain the non-modified text of the original Tweet.	keyword
objectType	String	Will always be "activity"	
object	Object	This object represents the source Tweet being posted or shared. For Original Tweets, this will contain a "note" object. For Retweets, this will contain an subset of original "activity" object to enable navigation to the source activity.	
postedTime	Date/ Time	The time the action occurred, e.g. the time the Tweet was posted in UTC format.	
favoritesCount	Integer	Valid for Retweets only. The number of users that have marked this Tweet as a favorite. Note that this is a "point in time" count. This number is updated as users Retweet the original Tweet, but user actions (favoriting the source Tweet or Retweets) do not generate activities.	
twitter_ filter_ level	String	Contains a string value denoting the relative importance of the activity. Valid values are "none", "low","medium" and "high" ("high is reserved for future use") A trending activity (and Retweets) will increase to "medium".	
twitter_lang	String	This string contains a BCP-47 code for the language as determined by machine learning algorithms at Twitter. If no language can be detected, it may be "und".	has:lang twitter_lang
retweetCount	Integer	Valid for Retweets only. The number of users that have shared this Tweet. Note that this is a "point in time" count, so only the most recent Retweet will have an accurate number.	
gnip	Object	This object contains data provided by standard and optional enhanced metadata such as matching_rules, Klout data, fully unwound URLs, and profile data.	
timestampMs	String	The timestamp of the user activity. Only applies to Compliance activities (where verb is delete, user_delete, user_undelete, scrub_geo, user_protect, user_unprotect, user_suspend, user_unsuspend, user_withheld and status_withheld).	

Actor Object

Property	Value	Description	Filtering operator(s)
id	String	A unique IRI for the Twitter user. When storing users, this should be used as the unique identifier or primary key.	
objectType	String	Will always be “person”.	
link	String	A Permalink for the Twitter user in URL form.	
displayName	Object	The user’s friendly name.	bio_name_contains
image	String	The url of the user's icon.	
summary	String	The user's bio.	bio_contains
postedTime	Date/Time	The time the action occurred, e.g. the time the Tweet was posted in UTC format.	
links	array of URLs	Format is href (string) and rel (string). “rel” is the display text for the link.	
location	Object	Geo data of user’s home	has:profile_geo has:profile_geo_locality has:profile_geo_subregion bio_location bio_location_contains profile_bounding _box profile_country_code profile_region profile_region_contains profile_subregion profile_subregion_contains
utcOffset	Number	Number of hours off of UTC for users profile “home” setting	
preferedUsername	String	Friendly display name	from
languages	array of strings		bio_lang
twitterTimeZone	String	Time zone selected (if any) by user	time_zone
friendsCount	Integer	Number of friends the user has	friends_count
followersCount	Integer	Number of followers the user has	followers_count
listedCount	Integer	Number of lists this user is on	listed_count
statusesCount	Integer	Number of status messages (Tweets) the user has created	statuses_count
verified	boolean	True for verified users	is:verified

Generator

Property	Value	Description	Filtering operator(s)
link	String	A Permalink to the client software download page.	
displayName	String	The name of the client that produced the Tweet.	

Provider

Property	Value	Description	Filtering operator(s)
objectType	String	Will always be “service”.	
link	String	A Permalink to Twitter’s home page	
displayName	String	Will always be “Twitter”	

inReplyTo

Property	Value	Description	Filtering operator(s)
link	String	URL of Tweet being replied to	retweets_of_status_id

Location

Property	Value	Description	Filtering operator(s)
objectType	String	May be“place” or “POI”	
displayName	String	The full name of the place. May be generalized to region (city, etc.)	place place_contains
name	String	Place.name from Twitter’s JSON format	
streetAddress	String	The street address of the place if available	
country_code	String	Expanded name of country	country_code
twitter_ country_ code	String	Two character representation of country	
link	String	A link to the full Twitter JSON representation of the place	
geo	Object	Bounding box of area Tweet was created	point_radius bounding_box

Geo (of Location object)

Property	Value	Description	Filtering operator(s)
type	String	type of geo - may be “point” or “Polygon”	
coordinates	Array	Double nested array of points that make up location (lat/lon)	

Geo (of Activity Object)

Property	Value	Description	Filtering operator(s)
type	String	type of geo - may be “point” or “Polygon”	
coordinates	Array	Array of points that make up location (lat/lon)	

Twitter_Entity

Property	Value	Description	Filtering operator(s)
hashtags	Object	Array of objects of hashtags in Tweet	#hashtag
symbols	Object	Array of objects of symbols in Tweet	
urls	Object	Array of objects of URLs in Tweet	url_contains
user_mentions	Object	Array of objects user mentions	has:mentions @mention

Hashtag & Symbol

Property	Value	Description	Filtering operator(s)
text	String	The tweet or symbol	see above
indicies	Array of integers	Start and Stop character substring locations of text in Body property	

URL

Property	Value	Description	Filtering operator(s)
url	Object	text of URL in Body (t.co...)	has:links url_contains
expanded_url	Object	Fully expanded URL after all redirects	url_contains
display_url	Object	First redirect of embedded URL	url_contains
indicies	Array of Integers	Start and Stop character substring locations of text in Body property	

User_Mention

Property	Value	Description	Filtering operator(s)
screen_name	String	Username mentioned - without "@"	@mention
name	String	Full name of account mentioned	
id	Long Integer	Account Id	
id_str	String	Account Id as a string	
indicies	Array of Integers	Start and Stop character substring locations of text in Body property	

Twitter_Extended_Entity

Property	Value	Description	Filtering operator(s)
media	Array of Objects	Collection of Media objects	has:media url_contains

Media

Property	Value	Description	Filtering operator(s)
id	Long Integer	Unique identifier for media	
id_str	String	Unique identifier as string	
indicies	Array of Integers	Start and Stop character substring locations of text in Body property	
media_url	String	URL of media (including HTTP reference)	
media_url_https	String	URL of media (including HTTPS reference)	
url	String	Original format of URL link as it appears in the message	
display_url	String	Not a true URL but string as it is displayed in the message	
expanded_url	String	Actual URL after redirects (includes HTTP/S)	
type	String	describes type of media - (photo is only type for now)	
sizes	Array of Size Objects	Size objects contain size name (small, medium, large, thumb), width and height dimensions, and sizing method (fit or crop)	

Gnip

Property	Value	Description	Filtering operator(s)
matching_rules	Array of Objects	All rules that match for a given activity	
urls	Array of Objects	All URLs referenced in body of message	
klout_score	integer	Klout score of posting user (actor)	
klout_profile	Object	Detailed information from Klout on user	
language	Single value Object	"value" parameter indicates 2 character representation of profile language	

Matching_Rules

Property	Value	Description	Filtering operator(s)
value	String	exact rule that matched for activity. Not included if using long rules	
tag	String	optional tag if set on rule	

URL (Gnip)

Property	Value	Description	Filtering operator(s)
url	String	Full URL referenced in Activity	
expanded_url	String	Fully resolved URL after all redirects	
display_url	Long Integer	Not a true URL, but how the URL is displayed	
indicies	String	location within body of URL string	
expanded_status	integer	HTTP response code of URL unwinding operation (200 = success)	

Klout_Profile

Property	Value	Description	Filtering operator(s)
topics	Object	Username mentioned - without “@“	
klout_user_id	String	Full name of account mentioned	
link	Long Integer	Account Id	

Topic

Property	Value	Description	Filtering operator(s)
klout_topic_id	String	Numeric identifier of Klout topuc	
displayName	String	Full description of Klout topic	
link	String	Klout topic URL	

ProfileLocation

Property	Value	Description	Filtering operator(s)
objectType	String	Notes granularity of profile location - “place”	
geo	Object	Geo point - center of “place”.	
address	Object	Detailed address information of profile	
displayName	String	Display ready version of Profile location	

Address

Property	Value	Description	Filtering operator(s)
country	String	Full name of country	
countryCode	String	Two character code of Country	
locality	String	City name or equivalent	
Region	String	State name or equivalent	
subRegion	String	County name or equivalent	

PowerTrack Rule Operators

Text Operators

Keyword
Matches a keyword within the body of an activity. The keyword is NOT surrounded by quotes. This is a tokenized match, meaning that your keyword string will be matched against the tokenized text of the activity body as described above. To match strings containing punctuation (e.g. coca-cola), symbol, or separator characters, you must use a quoted exact match as described below.

Gnip Rule	Match	No Match
gnip	I need to call gnip. I like #gnip. Check out @gnip.	#gniprocks
cola	I like coca-cola!	like cocacola!

has:hashtags
Matches Tweets that contain a hashtag.
WARNING: Use this operator with care. Used by itself, with no other limiting clauses, it can generate large amounts of volume. Currently, this will deliver double digit percentages of the firehose when used by itself.

Gnip Rule	Match
cat has:hashtags	My cat is too fat. #diet My cat just had kittens. #cute

#hashtag
Matches any activity with the given hashtag.
Also supports quoted hashtags, for matching “phrase tags” that contain whitespace characters.
This operator performs an exact match, NOT a tokenized match, meaning the rule “2012” will match posts with the exact tag “2012”, but not those with the tag “2012 election”
Note that the hashtag operator relies on Twitter’s entity extraction to match hashtags, rather than extracting the hashtag from the body itself. The description of how Twitter extracts entities can be found at http://dev.twitter.com/pages/tweet_entities.

Gnip Rule	Match	No Match
#politics	All posts tagged with "politics"	
#"2012 election"	All posts with the exact tag "2012 election"	
#boulderfire	posts containing "#boulderfire"	posts containing "#boulderfirefighters"

"text match"
Matches an exact phrase within the body of an activity. This is an exact match, and it is not necessary to escape characters with a backslash. For example, if matching something with a slash, use “one/two”, not “one\/two”.
Note that this is not a substring match, and includes a check for word boundaries at the ends of the quoted phrase. For a pure substring match, see the contains: operator below.

Gnip Rule	Match	No Match
“call gnip”	I need to call gnip, again I need to call gnip again call gnip	I called gnip call gnip (multiple spaces) call-gnip call_gnip

contains:
Substring match for activities that have the given substring in the body, regardless of tokenization. In other words, this does a pure substring match, and does not consider word boundaries.
Use double quotes to match substrings that contain whitespace or punctuation.

Gnip Rule	Match	No Match
contains:phone	Where is my phone? That's a telephone	Where is my fone?
contains:"he cat"	Where is the cat?	

"keyword1 keyword2"~N
Commonly referred to as a proximity operator, this matches an activity where the keywords are no more than N tokens from each other.
If the keywords are in the opposite order, they can not be more than N-2 tokens from each other.
Can have any number of keywords in quotes.
N cannot be greater than 6.

Gnip Rule	Match	No Match
"love boulder"~4	Love everything about my town Boulder. Boulder, I love living here.	I don't love hiking, but I really like to visit Boulder. Boulder is a place I love to visit.

User Operators

@mention
Matches any Tweet that mentions the given username. Note that this does not support user IDs as an argument. The username should not be quoted.
Note that the mention operator relies on Twitter's entity extraction to match mentions, rather than trying to extract the mention from the body itself.
The description of how Twitter extracts entities can be found here: http://dev.twitter.com/pages/tweet_entities

Gnip Rule	Match	No Match
@gnip	scored cool @gnip stuff at the meeting	I don't love hiking, but I really like to visit Boulder. Boulder is a place I love to visit.

from:
Matches any activity from a specific user. Can either be a username or MD5-hashed email.
For Twitter, the user ID can be used as well.

Gnip Rule	Match
from:mikesmith	All posted articles, comments, or likes from user mikesmith.
from:17200003	All posted articles, comments, or likes from user with this ID.

to:
Matches any activity that is in reply to a particular user.
In Twitter, the value must be the user's screen name or numeric ID (excluding the @ character).

Gnip Rule	Match
to:gnip	All posted articles, comments, or likes to user gnip.
to:16958875	All posted articles, comments, or likes to user 16958875.

retweets_of:
Matches tweets that are retweets of a specified user. Accepts both usernames and User IDs (NOT Tweet status ids)

Gnip Rule	Match
retweets_of:justinbieber	Any message that is a Retweet of a message created by justinbieber
retweets_of:6264412	Any message that is a Retweet of a message created by user id 6264412

bio_contains:
Matches tweets whose author's Twitter bio contain the given substring. To search for patterns with punctuation in them (i.e. start-up) enclose the search term in quotes.

Gnip Rule	Match	No Match
bio_contains:CEO	"CEO of ABC Corp"	"COO at DEF, Inc."
bio_contains:"Start-up"	"Start-up junkie"	"Software Engineer startup @Gnip"
bio_contains:"bieber"	"World's biggest @justinbieber fan"	"I love da biebs"

bio_name_contains:
Matches Tweets where the user's display name (not username) as specified in their bio, contains a given substring.

Gnip Rule	Match
bio_name_contains:"Mike"	(Any Tweets from a user whose said they were named Mike in their Twitter bio)

klout_score:
Matches Tweets whose author's Klout score (if, and only if, they have one) are within the given range. Klout scores are represented by integers from 1 to 100.
If a single number is specified, any score equal to or higher will match. For example, klout_score:50 will match any Tweet whose author's Klout score is 50 or higher.
Additionally, a range can be specified to match any scores in the given range. For example, klout_score:50..75 will match any Tweet whose author's score is equal to or greater than 50, but equal to or less than 75.
Note: Klout score data, and therefore the klout_score: operator, is only available via Gnip's Enriched output format.

Gnip Rule	Match
klout_score:75	Users with a klout_score of 75 or higher
klout_score:0..50	Users with a klout_score of 0 to 50, inclusive

klout_topic:
Matches Tweets where the Klout Topics provided for the Tweets author include the Topic name specified in the rule.
Note: Klout Topics data, and therefore the klout_topic_id: operator, is only available to Gnip's Premium Klout Enrichment subscribers.

Gnip Rule	Match
klout_topic:movies	"Movies" is one of the Klout Topics provided in Gnip's enrichment.

klout_topic_contains:
Matches Tweets where the Klout Topics provided for the Tweets author include the Topic name specified in the rule as a substring.
Note: Klout Topics data, and therefore the klout_topic_id: operator, is only available to Gnip's Premium Klout Enrichment subscribers.

Gnip Rule	Match
klout_topic_contains:music	Topics like "Musicians, Music Videos, Country Music"

klout_topic_id:
Matches Tweets where the Klout Topics provided for the Tweets author include the Topic ID specified in the rule.
The Klout Topic ID can be identified in the gnip.klout_profile.topics.link field for each Topic provided for a given user.
Its utility is generally interchangeable with the klout_topic rule by it may be a preferable method for some use cases.
Note: Klout Topics data, and therefore the klout_topic_id: operator, is only available to Gnip's Premium Klout Enrichment subscribers.

Gnip Rule	Match
klout_topic_id:8454101481033551211	messages where the klout_topic contains "Movies"

statuses_count:
Matches Tweets where the author has posted a number of statuses that falls within the given range.
If a single number is specified, any number equal to or higher will match.
Additionally, a range can be specified to match any number in the given range.

Gnip Rule	Match
statuses_count:1000000	messages where the sender has sent over 1 million Tweets

followers_count:
Matches Tweets where the author has a followers count within the given range.
If a single number is specified, any number equal to or higher will match.
Additionally, a range can be specified to match any number in the given range.

Gnip Rule	Match
followers_count:1000000	messages where the sender has over 1 million followers

friends_count:
Matches Tweets where the author has a friends count (the number of users they follow) that falls within the given range. If a single number is specified, any number equal to or higher will match.
Additionally, a range can be specified to match any number in the given range.

Gnip Rule	Match
friends_count:1000000	messages where the sender follows over 1 million users

listed_count:
Matches Tweets where the number of times the author has been added to Twitter User Lists falls within the given range. If a single number is specified, any number equal to or higher will match.
Additionally, a range can be specified to match any number in the given range.

Gnip Rule	Match
listed_count:1000000	messages where the sender is on over 1 million lists

is:verified
Deliver only Tweets where the author is "verified" by Twitter.
Can also be negated to exclude Tweets where the author is verified.

Gnip Rule	Match
dog is:verified	messages about dogs from verified users
cat -is:verified	messages about cats from unverified users

has:mentions
Matches Tweets that mention another Twitter user.
WARNING: Use this operator with care. Used by itself, with no other limiting clauses, it can generate large amounts of volume. Currently, this will deliver double digit percentages of the firehose when used by itself.

Gnip Rule	Match
has:mentions	"Did you hear what happened at @Gnip the other day?"

Location Operators

point_radius:[lon lat radius]
Matches against the Exact Location (x,y) of the Activity when present, and in Twitter, against a "Place" geo polygon, where the Place is fully contained within the defined region.
- Units of radius supported are miles (mi) and kilometers (km).
- Radius must be less than 25mi.
- Longitude is in the range of ±180
- Latitude is in the range of ±90
- All coordinates are in decimal degrees.
- Rule arguments are contained with brackets, space delimited.

Gnip Rule	Match	No Match
point_radius: [-105.27346517 40.01924738 10.0mi]	Tweets or Checkins within 10 miles of 17th and Pearl Street in Boulder, CO.	Tweets or Checkins outside more than 10 miles from 17th and Pearl in Boulder, CO

bounding_box:[west_long south_lat east_long north_lat]
Matches against the Exact Location (x,y) of the Activity when present, and in Twitter, against a "Place" geo polygon, where the Place is fully contained within the defined region.

- west_long south_lat represent the southwest corner of the bounding box where west-long is the longitude of that point, and south_lat is the latitude.
- east_long and north_lat represent the northeast corner of the bounding box, where east_long is the longitude of that point, and north_lat is the latitude.
- Width and height of the bounding box must be less than 25mi
- Longitude is in the range of ±180
- Latitude is in the range of ±90
- All coordinates are in decimal degrees.
- Rule arguments are contained with brackets, space delimited.

Gnip Rule	Match	No Match
bounding_box: [-105.301758 39.964069 -105.178505 40.09455]	Tweets or Checkins with Coordinates contained within a box drawn around Boulder, CO	Tweets or Checkins outside the box drawn around Boulder, CO

bio_location:
Matches Tweets where the user's bio-level location contains the specified keyword or phrase. This operator performs a tokenized match, similar to the normal keyword rules on the message body.
The user bio location is a non-normalized, user-generated, free-form string.

Gnip Rule	Match
bio_location:"boulder"	Boulder Boulder, CO Boulder Colorado Beautiful Boulder, CO

bio_location_contains:
Matches Tweets where the user's bio-level location contains the specified substring.
The user bio location is a non-normalized, user-generated, free-form string.
Warning: use of broad or common locations strings can result in the consumption of large volumes of data (e.g. a bio_location_contains:"MA" rule with hopes of matching all Tweets from Massachusetts, will also match "Alabama"). The addition of punctuation (e.g. ", MA" or ",MA") could help limit this data.

Gnip Rule	Match
bio_location_contains:"co"	Boulder, CO Boulder Colorado Beautiful Boulder, CO Conway, AR

time_zone:
Matches Tweets where the user-selected time zone specified in a user's profile settings matches a given string. These values are normalized to the options specified on a user's account settings page: [https://twitter.com/account/settings]

Gnip Rule	Match
time_zone:"Eastern Time (US & Canada)"	messages from users who have indicated the time zone in their profiles.

place:
Matches Tweets tagged with the specified location. Multi-word place names ("New York City", "Palo Alto") should be enclosed in quotes.

Gnip Rule	Match
place:"Rio de Janeiro"	messages tagged with the location of Rio de Janeiro

place_contains:
Matches Tweets where the tagged place/location contains a given substring.
Place names are semi-normalized by Twitter application but there can be many variations. A substring match allows you to easily match across variations.

Gnip Rule	Match
place_contains:"Boulder"	messages tagged with location "Boulder, NV", "Boulder, CO", "Boulder", "Boulder, Colorado, USA", etc.

country_code:
Matches Tweets where the country code associated with a tagged place/location matches the given ISO alpha-2 character code. Valid ISO codes can be found here: [http://en.wikipedia.org/wiki/ISO_3166-1]

Gnip Rule	Match
country_code:gb	messages tagged with the country of Great Brittan

has:geo
Matches Tweets that have Tweet-specific geo location data provided from Twitter. This can be either “geo” lat-long coordinate, or a “location” in the form of a Twitter “Place”, with corresponding display name, geo polygon, and other fields.
WARNING: Use this operator with care. Used by itself, with no other limiting clauses, it can generate large amounts of volume. Currently, this will deliver 1-4% of the firehose when used by itself.

Gnip Rule	Match
cat has:geo	messages tagged with geographic data with the token “cat” found

has:profile_geo
Matches Tweets that have any Profile Geo metadata, regardless of the actual value.

Gnip Rule	Match
cat has:profile_geo	Any Tweets that mentions the word “cat” and has Profile Geo metadata.

has:profile_geo_locality
Matches all activities that have a profileLocations.address.locality value present in the payload.

Gnip Rule	Match
profile_country_code:us has:profile_geo_locality	All Tweets with Profile Geo locations in the US that include city-level detail.

has:profile_geo_region
Matches all activities that have a profileLocations.address.region value present in the payload.

Gnip Rule	Match
profile_country_code:us has:profile_geo_region	All Tweets with Profile Geo locations in the US that include region-level detail (e.g. US states).

has:profile_geo_subregion
Matches all activities that have a profileLocations.address.subRegion value present in the payload.

Gnip Rule	Match
profile_country_code:us has:profile_geo_subregion	Tweets with Profile Geo locations in the US that include sub-region (county) level detail.

profile_bounding_box:[west_long south_lat east_long north_lat]
Matches functionality described for the standard bounding_box: operator, but only applies to geo-location data contained in the Profile Geo enrichment.

Gnip Rule	Match
profile_bounding_box: [-105.301758 39.964069 -105.178505 40.09455]	Profile Geo Enrichments with coordinates contained within a box drawn around Boulder, CO

profile_country_code:
Exact match on the “countryCode” field from the “address” object in the Profile Geo enrichment.
Uses a normalized set of two-letter country codes, based on ISO-3166-1-alpha-2 specification. This operator is provided in lieu of an operator for “country” field from the “address” object to be concise.

Gnip Rule	Match
profile_country_code:us	All Profile Geo Enrichments in the United States.

profile_region:
Matches on the “region” field from the “address” object in the Profile Geo enrichment.
This is an exact full string match. It is not necessary to escape characters with a backslash. For example, if matching something with a slash, use “one/two”, not “one\/two”. Use double quotes to match substrings that contain whitespace or punctuation.

Gnip Rule	Match
profile_region:"New York"	All Profile Geo Enrichments in New York state

profile_region_contains:
Matches on the “region” field from the “address” object in the Profile Geo enrichment.
This is a substring match for activities that have the given substring in the body, regardless of tokenization. Use double quotes to match substrings that contain whitespace or punctuation.

Gnip Rule	Match
profile_region_contains:carolina	All Profile Geo Enrichments in North or South Carolina (or other regions of the world containing the string "Carolina")

profile_locality:
Matches on the “locality” field from the “address” object in the Profile Geo enrichment.
This is an exact full string match. It is not necessary to escape characters with a backslash. For example, if matching something with a slash, use “one/two”, not “one\/two”. Use double quotes to match substrings that contain whitespace or punctuation.

Gnip Rule	Match
profile_locality:boulder	All Profile Geo Enrichments in ANY city named "Boulder"

profile_locality_contains:
Matches on the “locality” field from the “address” object in the Profile Geo enrichment.
This is a substring match for activities that have the given substring in the body, regardless of tokenization. Use double quotes to match substrings that contain whitespace or punctuation.

Gnip Rule	Match
profile_locality_contains:haven	All Profile Geo Enrichments in ANY city containing the substring "haven" including "New Haven," "West Haven," and "Lock Haven"

profile_subregion:
Matches on the “subRegion” field from the “address” object in the Profile Geo enrichment. In addition to targeting specific counties, these operators can be helpful to filter on a metro area without defining filters for every city and town within the region.
This is an exact full string match. It is not necessary to escape characters with a backslash. For example, if matching something with a slash, use “one/two”, not “one\/two”. Use double quotes to match substrings that contain whitespace or punctuation.

Gnip Rule	Match
profile_subregion:"San Francisco County"	All Profile Geo Enrichments where the subRegion is San Francisco County.

profile_subregion_contains:
Matches on the “subRegion” field from the “address” object in the Profile Geo enrichment. In addition to targeting specific counties, these operators can be helpful to filter on a metro area without defining filters for every city and town within the region.
This is a substring match for activities that have the given substring in the body, regardless of tokenization. Use double quotes to match substrings that contain whitespace or punctuation.

Gnip Rule	Match
profile_subregion_contains:jefferson	All Profile Geo Enrichments where the substring 'jefferson' appears in the subRegion (e.g. 'Jefferson County')

Language Operators

has:lang
Matches activities which Gnip has classified as any language.

Gnip Rule	Match
has:lang	messages where language has been classified

lang:
Matches activities that have been **classified by Gnip** as being of a particular language (if, and only if, the activity has been classified). Current languages supported are:

ar - Arabic da - Danish de - German el - Greek	en - English es - Spanish fa - Persian fi - Finnish	fr - French he - Hebrew it - Italian id - Indonesian	ja - Japanese ko - Korean nl - Dutch no - Norwegian	pl - Polish pt - Portuguese ru - Russian sv - Swedish	th - Thai tr - Turkish uk - Ukrainian zh - Chinese

It is important to note that each activity is currently only classified as being of one language, so AND'ing together multiple languages will yield no results. Also note that not every activity is classified as being of a particular language.

Gnip Rule	Match	No Match
lang:de	Liebe alles an meiner Stadt Boulder. Boulder , ich liebe , die hier leben.	Love everything about my town Boulder. Boulder, I love living here.

twitter_lang:
Matches Tweets that have been classified by Twitter as being of a particular language (if, and only if, the Tweet has been classified). According to Twitter documentation, the language code values 'will be valid BCP 47 language identifiers, and may represent any of the languages listed on Twitter's advanced search page, or "und" if no language could be detected.'
It is important to note that each activity is currently only classified as being of one language, so AND'ing together multiple languages will yield no results.

Gnip Rule	Match	No Match
twitter_lang:fr	"C'est un plaisir de vous rencontrer!"	"Nice to meet you!"

bio_lang:
Matches tweets where the user's bio-level language setting matches a given ISO 639-1 language code. Twitter does not support all languages in this list.
NOTE: This language setting simply changes the language which Twitter displays its UI text (it does not translate Tweet text). THIS IS NOT A LANGUAGE CLASSIFICATION. Customers have reported that this setting is often left in its default of English even when the Tweets an account is generating are in a foreign language.
We recommend its use in conjunction with Gnip's language classification operator (lang) rather than a standalone indicator of a user or Tweet's language.

Gnip Rule	Match
bio_lang:fr	Messages where sender has set the UI language to French.

Links Operators

has:links
This operators matches activities which contain links in the message body.

Gnip Rule	Match	No Match
cat has:links	Here's a picture of my cat: bit.ly/cat Adopt a cat at http://spca.org/cats	I don't love hiking, but I really like to visit Boulder. Boulder is a place I love to visit.

url_contains:
Matches activities with URLs that literally contain the given phrase or keyword. URL encodings are not encoded at this time. To search for patterns with punctuation in them (i.e. google.com) enclose the search term in quotes.
NOTE: If you're using Gnip's Enriched output format, we will match against Gnip's expanded URL as well.

Gnip Rule	Match
url_contains:microsoft	cool link http://microsoft.com
url_contains:"google.com"	cool link http://www.google.com
url_contains:"/apps/ipad"	cool link http://foo.com/apps/ipad/a.htm
url_contains:"iphone"	cool link http://bit.ly/123455 (if the expanded bit.ly URL contains "iphone")

Attribute Operators

source:
Matches any Tweet generated by the given source application. The value must be either the name of the application, or the application's URL. Cannot be used alone.

Gnip Rule	Match	No Match
cat source:web	cool cat (if the Tweet was created at twitter.com)	cool cat (if the Tweet was created from an iPhone)

sample:
Returns a random sample of activities that match a rule rather than the entire set of activities. Sample percent must be represented by an integer value between 1 and 100. This operator applies to the entire rule and requires any "OR'd" terms be grouped. The sample operator has precedence (occurs before the other operators).

Gnip Rule	Returns
cat sample:10	all tweets with the keyword "cat" within a 10% sample of the firehose
(dog OR cat) sample:25	all tweets with the keyword "cat" or "dog", within a 25% sample of the firehose

has:media
Matches Tweets that contain a media url classified by Twitter, e.g. pic.twitter.com.
WARNING: Use this operator with care. Used by itself, with no other limiting clauses, it can generate large amounts of volume. Currently, this will deliver double digit percentages of the firehose when used by itself.

Gnip Rule	Match
has:media cat	"check out this pic of my cat http://www.catscans.com/pic/flippingcat.gif"

is:retweet
Deliver only explicit Retweets that match a rule. Can also be negated to exclude Retweets that match a rule from delivery and only original content is delivered.
NOTE: This operator looks only for true Retweets, which use Twitter's Retweet functionality. Quoted Tweets and Modified Tweets which do not use Twitter's Retweet functionality will not be matched by this operator.

Gnip Rule	Match
dog is:retweet	retweets of messages about dogs

retweets_of_status_id:
Deliver only explicit Retweets of the specified Tweet.
Note that the status ID used should be the ID of an original Tweet and not a Retweet. If extracting the ID of an original Tweet from within a Retweet for this purpose, look in the object.id field in Activity Streams format.

Gnip Rule	Match
retweets_of_status_id:365697420392280064	retweets of message 365697420392280064

in_reply_to_status_id:
Deliver only explicit replies to the specified Tweet.

Gnip Rule	Match
n_reply_to_status_id: 365697420392280064	Replies to the Tweet with status 365697420392280064

Frequently Asked Questions

"How fast do Tweets arrive after being created?"
Gnip can deliver Tweets in two formats - either Original format (directly from Twitter) or Activity Format as described in this document. The enrichments available are only delivered in Activity Format. Tweets in Activity Format are delivered ~ 15 seconds after creation. Original Twitter format streams deliver Tweets in ~2 seconds after creation.

Can we set up multiple streams with PowerTrack?
While this is technically possible, it will impact pricing. The best practice for using PowerTrack would be to use rule tags as a way to differentiate customers or use cases. Once the activity is read from the stream, the matching_rules value can be inspected and used to steer the individual activity to one or more applications/customer processes, etc.

How can I tell the difference between a Tweet, a ReTweet, a ReplyTo and a TweetQuote?
Using a combination of the verb, inReplyTo and object fields, the exact type of message can be determined. The following table highlights the differences:

Type	Verb	inReplyTo	Object field
New Tweet	post	null / doesn't exist	Points to self. Activity.Link same as Activity.Object.Link
ReTweet	share	null / doesn't exist	Contains original Tweet object
Reply	post	Link to original Tweet	Points to self. Activity.Link same as Activity.Object.Link
TweetQuote	post	null / doesn't exist	Object.link points to original Tweet

I have more questions! How can I get help?

On the web:
Our support web site is full of useful information: support.gnip.com

Pre-sales support:
Please email gnip-sales-engineering@twitter.com with questions or contact your existing Account Executive to arrange a meeting or conference call.

Existing customers: (post trial, signed contract)
Please contact Support at support@gnip.com