**LinkUp “RAW 2.0 Enhanced” Change Document**

Contents

[Purpose: To outline the changes from RAW1.0 to RAW2.0 Enhanced 2](#_Toc68262649)

[Summary: 2](#_Toc68262650)

[RAW: 3](#_Toc68262651)

[A look at the old files vs. the new. 3](#_Toc68262652)

[Current R2E vs. R1 raw files and schemas. 3](#_Toc68262653)

[R2E Raw Files R1 raw files 3](#_Toc68262654)

[Reposts! 6](#_Toc68262655)

[Last\_checked and created 7](#_Toc68262656)

[Descriptions Cleaned up! 7](#_Toc68262657)

[“Removed” Company Scrapes 7](#_Toc68262658)

[Scraper “Changed” 7](#_Toc68262659)

[Added the SOC/ONET code 8](#_Toc68262660)

[Added the job URL 8](#_Toc68262661)

[Adjusted the Overall Scrape Cycle 8](#_Toc68262662)

[We now enforce COMPLETE company level scrapes before putting in production 8](#_Toc68262663)

[We added Factset PIT identifiers 8](#_Toc68262664)

[We have an enhanced “Downloader” 8](#_Toc68262665)

[Analytics! The Fun Part! 9](#_Toc68262666)

[Current R2E vs. R1 analytics files and schemas. 9](#_Toc68262667)

[Initial Aggregates 11](#_Toc68262668)

[Hash Level Analytics 11](#_Toc68262669)

[File Delivery 12](#_Toc68262670)

[Timing 12](#_Toc68262671)

[AWS S3 13](#_Toc68262672)

[Appendix 1: RAW2 RAW data supplementary charts 14](#_Toc68262673)

[BAD Daily hash – *where company\_id or dates are malformed* 14](#_Toc68262674)

[Reposts – 14](#_Toc68262675)

[Company\_ids scraped that had active jobs 16](#_Toc68262676)

[Daily jobs\_log file size in MB 16](#_Toc68262677)

[Total jobs\_log file size in GB 17](#_Toc68262678)

[Daily jobs\_base file size in MB 17](#_Toc68262679)

[Bad “New” created dates. Multiple created dates for same hash 17](#_Toc68262680)

# Purpose: To outline the changes from RAW1.0 to RAW2.0 Enhanced

# Summary:

RAW2.0 Enhanced (R2E) provides many game changing enhancements versus RAW1 (R1). We have:

* Re-envisioned the entire dataset and analytics to streamline usage, data footprint, and usability. Dramatically improved ETL.
* Made the dataset completely *point in time* (PIT), with each day as you would have seen the data on that day. This is a true job level PIT history.
* Separated mutable from immutable data to streamline files, ingestion, ETL.
* Added a detailed and complete “scrapelog”.
* Removed the noisy “ad” and “hosted” jobs which we included but recommended not using in R1.
* Added Factset identifiers to supplement our previous Refinitiv identifiers.
* Simplified reference data, to 1 user friendly point in time file
* Greatly simplified the file structure. Eliminated file “bloat”.
* Simplified file ingestion significantly
* Dramatically reduced file storage footprint
* Fixed the systemic “reposting” issue.
* Added a simple python downloader.
* Fixed “bad” raw data, including malformed state and country fields, and null date fields.
* Fixed bad “new” created dates, where a hash would show up with 2 or more created dates.
* We remove FTP access, shifting to AWS S3 as it offers far superior access and automated sync capabilities
* AWS Glue
* Removed duplicate ‘hash’
* Deal with hash that have multiple different “created” dates over time. Should never happen.
* Added standardized 2010 BLS SOC classification codes and removed our proprietary “category\_id”
* Added the job “url”
* Added a significant amount of PIT analytics and metadata.
* Added a simple python downloader

# RAW:

Let’s start with the actual RAW jobs data and descriptions, the “heart” of our data and analytics.

R1 had a jobs file, and a descriptions file. We provided these files starting 2016-01-23, when we started to distribute the files. But our daily data history goes back to 2007-08-03. While the data is accurate, in R1 we lost the Point In Time (PIT) nature of it. The changes described below create a perfect PIT representation of the raw data, and greatly improve usability, veracity, and data credibility while expanding the overall possibilities for analytics creation.

## A look at the old files vs. the new.

R2E is formatted completely in the ultra slim, fast, and flexible [Apache Arrow “parquet” format](https://parquet.apache.org/), built for Big Data. R1 is in csv and xml.

## Current R2E vs. R1 raw files and schemas.

#### Fields in GREEN are added, in RED are deleted:

## R2E Raw Files R1 raw files

|  |  |
| --- | --- |
| **jobs\_base-** This is an enhanced “jobs” table, which is just the *immutable* data for a given hash, with some important modifications. It exists as a single file partitioned by ‘created\_pit (see below)’. It is pure PIT, all you do is ingest the file once, and add the created\_date you need every day.   * jobid – the jobid is a “smart” identifier.  The first 8 digits are the created date, *point in time*, meaning the date that the client would have seen that the job was createdà the actual production date (see below “created\_pit”).  The next 5 digits are the company\_id.  The final 5 digits is the specific number of the job that was created that day for that company\_id.  For example 202006090000100001 would be the jobid for Target for the first job that was created on 2020-06-09.  While “hash” is random, this identifier has information and therefore value, flexibility and simplicity. * repostFlag – 0 or 1 based on if a repost or not * company\_id * title * city * state * zip * country * created * createdPIT (this is the actual PIT production created date, used in the jobid, and to fix the created and production mismatch issue.) * parenthash (this is the parent “hash” (see the “hash” in the jobs schema on the right) for the jobid.  With this we can deal perfectly with all reposts) | **jobs**- This is the primary raw file which shows 1 record for each unique job, which each records changes over time. Some fields are mutable like “last\_checked”, which creates the PIT issue.   * Hash – the unique hash assigned based on unique URL of the job post. Added as parenthash in the jobs\_base file to the left * company\_id * company\_name – moved to mutable PIT reference file * title * city * state * zip * country * created * last\_checked – captured in PIT jobs\_log below * last\_updated * is\_ad * linkup\_hosted * linkup\_hosted\_historical * category\_id * category\_name * unmapped\_location |
| **jobs\_log-** With the enhanced *company\_scrape\_log (see below)*, and the *jobs\_base* as a foundation, we create a hash level ***true*** point in time record, all the way back to day 1.  So we would be able to look at 2007-08-03, and see the “daily” view on that date.  **The result though is a perfect PIT dataset, at the atomic “hash” level.**  The dataset is **approaching 5 BILLION records**!  BUT, because of how it is constructed it, it requires only ~17GB of storage space. Pull the whole file up to date, or just select the “scrape\_date(s)” you want.   * jobid – see above * scrapeDate – the PIT scrape date for the job * addremoveflag - The addremoveflag is also SMART!!!  It can be either 1,0,-1, where 1 means the jobid was created (the PIT created date as you would have seen in production-historically many jobs would have a created date of the day before you actually saw the job in our production process. This is true PIT), 0 means it was active and checked (this replaces ALL of the historical possible “last\_checkeds” in history, -1 means it was deleted (the PIT deleted date). | None |
| **scrapelog**- this is a great meta data enhancement which shows at the company\_id level each day the company was scraped, whether the company scraper was “changed”.   * company\_id, * actionDate, * actionType. (i.e. “scraped”, or “scrape\_changed” etc.) | None |
| **descriptions-**   * hash * description * hash1 – this is so you can iterate through this large data set easily in the file/partition. Simplifies processing * created\_pit – this is so you can pull specific days or just add a new day. Simplifies processing | **descriptions-**   * hash * description |
| **reference-** this is a great enhancement where we consolidate all of our reference data, PIT. This new file allows us to remove the bloat from the analytics files, as well as add new sources of identifiers at will, such as we did with Factset. This parquet is partitioned by refsource (i.e. Refinitiv/Factset/LinkUp/SmartMarketData etc.) as well as reftype (i.e. ticker, cusip, sedol, LEI, PermID etc.) for a fast and easy way to grab all at once, or just what you want.   * company\_id * refsource – the source of the data * refsource\_id – the primary identifier used by the refsource * reftype – the type of reference data – i.e. URL, or company\_name, sedol, or ticker etc. * start\_date – the start date of the mapping * end\_date – the end date of the mapping * value – the value (i.e. ‘AAPL’) * thrudate – the date that the data was added to the file | Previously we interspersed some reference data in “pseudo” PIT in the “company\_daily” and “company\_summary” files. While good, it was not scaleable. |
| **auxiliary- –** this is a file to hold auxiliary “helper” tables, like the 2010 and 2018 SOC code descriptions for example which are already posted. Wholly scaleable and simple to navigate. |  |

##### *File Sizes*

|  |  |
| --- | --- |
| **jobs\_base** size is <15GB zipped, and that is IT. Every day you just consume the new data which is easily ingested via the created\_pit partition, and is very small (MBs) | **jobs** - All Dailies are ~500GB zipped, and only go back to 2016. This file set grows ~300MB per day ZIPPED. These do include descriptions  All jobs archives are ~520GB, and only go back to 2016 as well. These archives grow almost 1GB per month. So over 1TB of data stored here, growing 10GB per month. |
| **jobs\_log** size is < 20GB zipped and that is IT. Every day you just consume the new data which is easily ingested via the scrape\_date partition, and is very small (MBs) | None |
| **scrapelog** size is inconsequential at 150MB growing very little per day | None |
| **descriptions** size is around 70GB zipped, total. Adding just MBs per day. No archives, deltas, etc. | **descriptions** size is included in the above “jobs dailies”, **but has its own archive, which represents ~3.2TB of data, adding 125GB zipped per month!** |
| **reference** size is inconsequential at <10MB | Added bloat to analytics files previously |
| **auxiliary** size is inconsequential |  |

The end result is the following RAW data file profile:

*R2E Files (all parquet)* *R1 Files (csv and xml)*

|  |  |
| --- | --- |
| jobs\_base | linkup\_raw\_daily\_yyyy-mm-dd.tar.gz |
| jobs\_log | linkup\_raw\_daily\_yyyy-mm-dd.csv |
| descriptions | linkup\_raw\_daily\_yyyy-mm-dd\_descriptions.xml |
| scrapelog | linkup\_raw\_yyyy-mm-dd.tar.gz - archive |
| reference | linkup\_job\_descriptions\_yyyy-mm-dd.tar.gz - archive |
| auxiliary | none |
| #files/size footprint/daily growth – **6/~100GB/<100MB** | **1000’s/~4+TB/~11GB** |

## Reposts!

What is the repost issue? Historically, on 2016-01-23, we saw the raw jobs data for 2010-01-01 as it was on 2016-01-23. We had no record backwards of what you would have seen “point in time”. While differences are fairly insignificant, there is a systemic upward “drift” in job counts over time, based on the analytics and data capture methodology, due to companies “reposting” jobs they had previously removed for a period. For instance, company XYZ may have 10 jobs active on 2010-01-01. On the next “scrape” on 2010-01-02 they have 9 jobs, having removed 1. There are 10 scrape days where no changes occurred, so on 2010-01-12 we still show 9 jobs, and the time series for the period would show 10, then 9,9,9...9. Now company XYZ adds back the EXACT SAME JOB (based on the unique URL) on 2010-01-13. For 2010-01-13 we will show 10 jobs again. *BUT*… the time series will now adjust back that that job existed for ALL the days, so it would look like 10,then 10,10,10…10. A “drift” updward of 1 job.

Again, NOT a PIT representation of what we knew on a specific day. We fix this nuance and many other things in R2E.

We eliminated the repost issue by capturing when companies remove jobs, and if they add back, creating a “new” job – that still references the original “hash”. This is accomplished with the jobid, and the parenthash in the jobs\_base\_file. In doing this, we eliminate the “drift” that occurs non-PIT. We also eliminate the problem of having a deleted date be NULL, then have a date, then suddenly be NULL again. Our goal is to capture when companies have jobs actually posted. This modified look gives you just that.

## Last\_checked and created

In R1 we looked at the last\_checked (the date we last scraped and saw that the job existed), we then inferred a “deleted date” based on the “next” time the company\_id was scraped and the job did not exist. We then used the deleted date to calculate the time a job was active (>= the created date and < the deleted date). This gave the best possible idea of when a job was posted based on the information we have. But there were scenarios where last\_checked could be say at 11:00PM on day t, but the production file was for t+1. We fix this and adjust all the dates to their relevant “production” dates – the date YOU would have known that the job was scraped. With this we get accurate representation of PIT scrapes, as well as delete dates, as represented in the scrapelog and further in the job\_log. We do this for the created\_date as well. So, as an example, company XYZ is scraped from 11:00PM on 2020-01-01 to 1:00AM 2020-01-02. The created for a job here can be 2020-01-01, while the last\_checked can be 2020-01-02. This created mismatch errors which we adjust in RAW1, but here we need no adjustment, it is true. While the adjustments on edge are relatively insignificant, they do make the data “truer”.

## Descriptions Cleaned up!

The captured Description unstructured text can at times get a little messy. There are instances of “runaway” records where the text fields are giant, representing a clear error. While we try hard to deal with this in the normal scrape operations, some slip through. We deal with this in R2E by cutting them off, and limiting them to a reasonable size.

## “Removed” Company Scrapes

Here we capture company\_id level scrapes that have been “removed” from production, and we deal appropriately with removing jobs for these from the analytics post the removal. In R1, these jobs could persist in the analytics because we had no “next scrape” to infer a delete date. You will see these in the scrapelog described above.

## Scraper “Changed”

We identify when we make a change to the scrape code, as an addition to the scrapelog that can be used to help verify a “strange” change in the analytics.

## Added the SOC/ONET code

In R2 we have mapped all our jobs to the [2010 Bureau of labor Statistics SOC CODE](https://www.bls.gov/soc/), and the enhanced [O\*Net](https://www.onetonline.org/) code. This replaces our limited proprietary category\_id. In R2E this is available in the **analyticsEnhancedJob** file described below. We expect to add the 2018 codes as well, and various levels of salary data connected to this code in the near future. The beauty is, when we add, it will just be a new field in the **analyticsEnhancedJob** file. In R2 SOC code was available in the ‘daily jobs’ file, thus was repetitive. Also, the R2 format was unscalable (i.e. if you wanted to add SOC Code for 2018 you bloat the daily file even for those that don’t care).

## Added the job URL

In R2 we have added the job level unique URL and in R2E it is available in the **analyticsEnhancedJob** file as well. This can be used as a verification tool among other things. In R2 this was also in the ‘daily jobs’ file, again more repetitive bloat.

## Adjusted the Overall Scrape Cycle

In R1 we had a daily scrape cycle that went from roughly 2:00PM CT to 2:00PM CT. We have adjusted this to 12:00AM UTC to 12:00 UTC for better/easier date matching.

## We now enforce COMPLETE company level scrapes before putting in production

Historically, if we scraped a company and the scrape overlapped 2 daily scrape cycles, we would publish some jobs on the first day, and some jobs on the next day. Now, if a company level scrape is NOT complete, the jobs get pushed to the day the company level scrape is complete.

## We added Factset PIT identifiers

We have added Factset identifiers in addition to Refinitiv. We provide the ticker, sedol, cusip and isin…

## We have an enhanced “Downloader”

We have modified our somewhat rudimentary python downloader in R1 to a python downloader for R2E. All you need to do is enter the directory you want the data, enter your credentials, pick the files, and go. You can also adjust it to capture just the current days data.

## Analytics! The Fun Part!

With this new Gold Copy, job level, PIT raw data, we can do some great things with analytics. Let’s break down the files as we did for the raw data above. These files are also all in parquet, but are designed to be extremely flexible to consume. They also allow us to provide complete PIT looks of the analytics all the way back to 2007, whereas R1 only allowed for a “pseudo” PIT look back to 2016, and R2 had no part time analytics and only analytics in general back to November 2019. If you like data, buckle up!

### Current R2E vs. R1 analytics files and schemas.

#### Fields in GREEN are added, in RED are deleted:

##### R2E Analytics Files R1 Analytics Files

|  |  |
| --- | --- |
| **analyticsCore-** this file will contain the core analytics for any aggregate (company\_id, ticker, portfolio, GIC industry, Country, State etc.). Sweet and simple. Each aggreagate is the top level, and is given a “smart” code, which links to a table. You simply look at the table description, and download the “smart” code to get all the available aggregate data.   * jobsdate * active * created * removed * durationActive – the average length of time that the current active jobs have been posted for the aggregate * durationClosed – the average length of time that the “removed” jobs *were* posted | Company\_daily/monthly  Bucket\_daily/monthly (ticker level)  Macro daily/monthly  USMacro daily/monthly  USMacro10000 daily/monthly  USMacro\_xbrks daily/monthly  State daily/Monthly  Each of the above files was provided with full history, as well as full deltas (to account for reposts), as well as single day deltas (to create pseudo PIT) in both daily and monthly versions, each and every day. A summary of the fields:   * jobsactive * jobsactive\_n – now in analyticsEnhanced see below * ajnet * normalizedflag -now in analyticsEnhanced * bigdrpbrksflag – redundant * cvbbrksflag * jobsremoved * jobscreated * thrudate * ticker\_main * RCP Reuters Editorial RIC * Immediate Parent Reuters Editorial RIC * Ultimate Parent Reuters Editorial RIC * Derived Parent Reuters Editorial RIC * Derived Ultimate Parent Reuters Editorial RIC * ID\_BB\_SEC\_NUM\_DES * Postlength/2 * verified\_ds1 * verified\_ds2 * verified\_ds3 * verified\_ds4 * verified\_ipo * coids – see analyticsEnhanced * coids\_rj – see analyticsEnhanced * coids\_cj– see analyticsEnhanced * score\_desc– see analyticsWord below * score\_title– see analyticsWord * const– see analyticsEnhanced * constcnt– see analyticsEnhanced * ajindex – see analyticsEnhanced for enhanced indices * aji\_pc1 * coids\_cju * cjindex * cji\_pc\_1 * coids\_rju * rjindex * coids\_aju * ajnetindex * ajneti\_pc\_1 |
| **analyticsEnhanced-** this file provides secondary enhanced analytics   * jobsdate * constituentsActive * constituentsActiveCount * constituentsAdded * constituentsAddedCount * constituentsRemoved * constituentsRemovedCount * indexEqualWeighted (divisor adjusted) * indexJobsWeighted (divisor adjusted) * durationModified1 * rank * rankPer(%) * Much more coming! | None (some in various forms in fields listed above) |
| **analyticsFirst (coming soon)-** This file allows us to highlight “firsts” such as  “First part-time job” or “first SOC code” or “first Country”.  A client can quickly determine a company, ticker, portfolio, industry, any aggregate’s first day that they were in a City, State, Zip, Country, specific SOC code etc….   * jobsdate * firstType * value | None |

This format offers a flexible, scalable, and concise way to access ANY aggregates that we create. The footprint is a mere percentage of the current analytics files in R1. You access the aggregate (company, ticker etc.), and have access underneath it to all of the above files. You can select what you want. All the files are 100% PIT as we would have known on any given date. So no delta files. No PIT files. No archives. Just grab the file, and update based on the jobsdate “partition” that you need. In fact, we are considering offering portfolio level aggregate services to our customers. This is a massive improvement over the solid R1 analytics offering.

### Initial Aggregates

* MACRO
  + Company\_id
  + City
  + State
  + Zip
  + Country
  + Parttime (coming)
  + Fulltime (coming)
  + WorkFromHome (coming)
  + Ticker (RIC)
  + Ticker (Factset)
  + Sedol (Factset)
  + Naics sector
  + Naics subsector
  + Naics industry group
  + Naics industry
  + Naics sub industry
  + Trbc economic sector (requires Refinitiv License)
  + Trbc business sector (requires Refinitiv License)
  + Trbc industry group (requires Refinitiv License)
  + Trbc industry (requires Refinitiv License)
  + Soc major
  + Soc minor
  + Soc broad
  + Soc detailed
* USMacro1000
  + With all the breakdowns in Macro above

### Hash Level Analytics

To allow us to scale our hash level analytics, we have created the *analyticsEnhancedJob* file, for job level analytics.

|  |  |
| --- | --- |
| **analyticsEnhancedJob -** This file allows for us to add any level of information at the hash level. This file replaces the ‘hash’ file in R1   * jobid, * analyticType * value   Where the analyticType can be:   * url * soc\_2010 * soc\_2018 * soc\_2019 * part\_time * soc2010 pit salary * soc2010 pit salary by region * soc2010 pit salary by industry * soc2018 pit salary * soc2018 pit salary by region * soc2018 pit salary by industry * soc2019 pit salary * soc2019 pit salary by region * soc2019 pit salary by industry * …..much more | This file replaces the ‘hash’ file in R1, which was GBs per day.  (some like soc, url are found duplicative in the R1 jobs file) |

## File Delivery

### Timing

R2 RAW files are typically available around 12:30AM UTC. The R2E RAW data and reference files will be available roughly 45 minutes after the original R2 RAW files are posted.

The R2 company and ticker analytics are typically available around **7 hours** after the R2 RAW files, or 7:30AM UTC. The R2E Analytics are available within roughly **3 hours**, or 3:30AM UTC.

* + scrapelog ~9:30PM EDT/~8:30PM EST (1:30AM UTC)
  + jobs\_base ~10:30PM EDT/~9:30PM EST (2:30AM UTC)
  + jobs\_log ~10:30PM EDT/~9:30PM EST (2:30AM UTC)
  + analyticsEnhancedJob ~10:30PM EDT/~9:30PM EST (2:30AM UTC)
  + descriptions ~11:00PM EDT/~10:00PM EST (3:00AM UTC)
  + reference ~11:00PM EDT/~10:00PM EST (3:00AM UTC)
  + analyticsCore ~12:30AM EDT/~11:30PM EST (4:30AM UTC)

## AWS S3

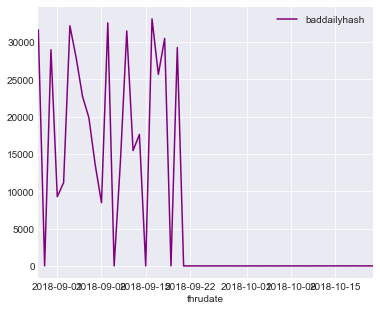
All R2E files are available on AWS S3, are AWS GLUE ready, and can be either downloaded locally or synced with an existing AWS bucket. The file topology is quite simple with the following “directories”, where extraction is merely updating the directory. You also have the option of extracting JUST certain parquet “partitions”:

* jobs\_base
* jobs\_log
* descriptions
* reference
* scrapelog
* auxiliary
* analyticsCore
* analyticsEnhancedJob

# Appendix 1: RAW2 RAW data supplementary charts

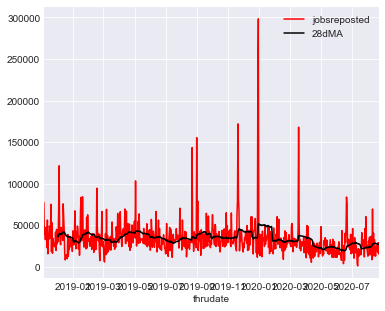
## BAD Daily hash – where company\_id or dates are malformed

There was a period in 2018 when there were daily files with sizeable “bad” rows.

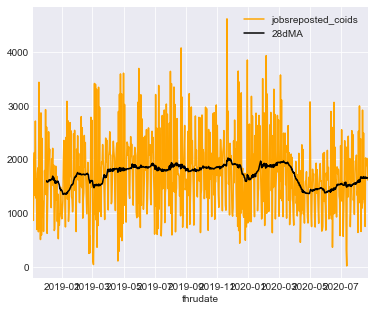


## Reposts –

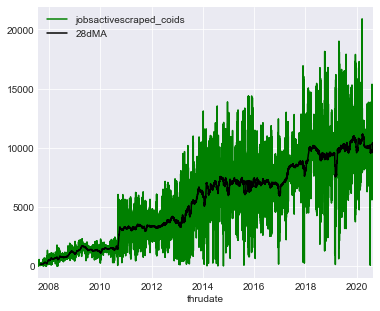
Deletes occur when a company is scraped, and a job that was active the previous scrape is not there. A repost happens when that same job is “reposted”, meaning in a subsequent scrape of the same company, it appears again. Reposts really kick in in the dataset in 2018.



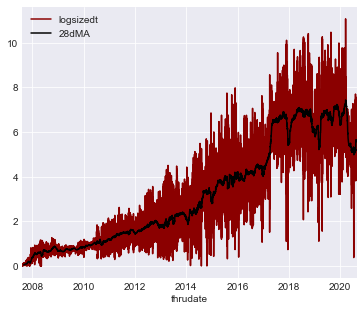
And the # of company\_ids with reposts:



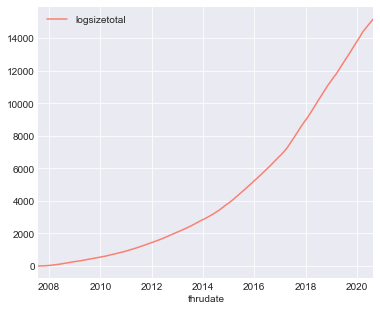
## Company\_ids scraped that had active jobs



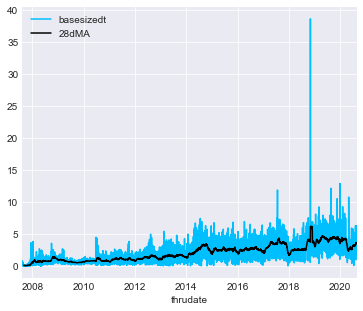
## Daily jobs\_log file size in MB



## Total jobs\_log file size in GB



## Daily jobs\_base file size in MB



## Bad “New” created dates. Multiple created dates for same hash