Info [1]	Name	Homepage	Source Code	Storage Engine	Language	Bytes/point (adve	Bytes/point (wors	Dimentions/Tags	Clustering	Comments NOTE: YOU CAN COMMENT (ADD SUGGESTIONS) THIS SHEET. For more information about the data - see the bottom of the sheet
	Atlas	https://github.com/Netflix/atlas/wiki/Overview	https://github.com/Netflix/atla	Custom	Scala			?	Yes	In-memory, multi-dimensional
	BigGraphite	https://github.com/criteo/biggraphite	https://github.com/criteo/bigra	Cassandra	Python	16	24 [4]	No [5]	Yes	It's design goals were set with whisper in mind, but based on cassandra
	CitusDB	https://github.com/citusdata/citus	https://github.com/citusdata/c	PostgreSQL	C	[6]	[7]		Yes	Yes [8]
	graphite-clickhouse	https://github.com/lomik/graphite-clickhouse	https://github.com/lomik/grap	Clickhouse	Golang	2 [9]	6,5 [10]	Partial***** [11]	Yes	Bytes/point - depends on nature of the data. Syntetic - 5-ish, real-world - 6-ish. Graphite compatibility layer with partial support for tags: https://github.com/lomik/carbon-clickhouse and https://github.com/lomik/graphite-clickhouse
	graphouse	https://github.com/ClickHouse/graphouse	https://github.com/ClickHou	Clickhouse	Java	5,4 [12]	6,5 [13]	No	Yes	Yandex's own graphite-compatible storage implementation: https://www.percona.com/live/17/sessions/clickhouse-time-series-storage-graphite
	Cortex	https://github.com/cortexproject/cortex	https://github.com/weaveworl	Custom	Golang					Long-term storage for prometheus
	Druid	http://druid.io	https://github.com/apache/dru	HBase	Java			?	Yes?	More an aggregator that can send data to some long term storage, not a proper TSDB or even DB.
	ElasticSearch	https://www.elastic.co/	https://github.com/elastic/elas	Apache Lucene	Java	22 [14]	98 [15]	Yes	Yes	
	FiloDB	https://github.com/filodb/FiloDB	https://github.com/filodb/FiloD	Spark + Cassand	Scala			?	Yes?	Requires Spark
	Gnocchi	https://gnocchi.xyz/	https://github.com/gnocchixyz	Ceph/S3/Redis/S	Python	6,25 [16]	9 [17]	Yes	Yes	Developed by OpenStack guys. Sounds good on paper. Uses Ceph/S3/Swift API to store time-series data. Requires MySQL or PostgreSQL for tags
	Heroic	https://spotify.github.io/heroic/#!/index	https://github.com/spotify/her	Cassandra	Java			Yes	Yes	
	InfluxDB	https://influxdata.com/	https://github.com/influxdata/	Custom	Golang	2 [18]	3 [19]	Yes	Commercial	Sounds good on paper. Have a long history of performance and stability problems. Still known for very long startup time for databases >100GB (as of 1.2 it might take several hours to restart it) @May 2017
	KairosDB	https://kairosdb.github.io/	https://github.com/kairosdb/k	Cassandra	Java	10 [20]	12 [21]	Yes	Yes?	Have graphite-compatible input
	m3db	https://github.com/m3db/m3db	https://github.com/m3db/m3db	Custom	Golang	1,4 [22]	1,4 [23]	Yes	Yes	TSDB used at Uber. http://www.devopspro.ru/wp-content/uploads/2016/11/Mantas-Klasavicius-Metrics-at-scale-@UBER.pdf - Gorilla-based compression, clustering support, used in Production at Uber (but that's a small piece of infrastructure)
	metrictank	https://github.com/raintank/metrictank	https://github.com/raintank/m	Cassandra	Golang	1,3 [24]	3 [25]	Yes (with caveats)	Yes [27]	Gorilla-based compression, limited clustering for metriktank, proper on cassandra. Can't overwrite history https://github.com/raintank/metrictank/issues/511. Currently can store data with tags, but can't query them
	NetData	https://github.com/firehol/netdata/	https://github.com/firehol/net	???	C			No?	No	Monitoring agent with a simple storage inside. Seems to be suitable over small setups or as an a collector
	NewTS	https://opennms.github.io/newts/	https://github.com/OpenNMS/	Cassandra	Java			Yes?	Cassandra-base	Have Graphite-compatible input
	open falcon	http://open-falcon.org	https://github.com/open-falco	Custom	Golang			Yes	Yes	advertised as suitable for time series. crazy idea in practice. Inspired by RRDTool
	OpenTSDB	http://opentsdb.net/	https://github.com/OpenTSDI	HBase or Cassar	Java	12 [28]	39 [29]	Yes	Yes	
	Prometheus	https://prometheus.io/	https://github.com/prometheu	Custom + LevelD	Golang	1,3 [30]	3,3 [31]	Yes	Federation	Compression, Not suitable for long term storages@May 2017. Version 2.0 have Long Term Storage API and can use other databases as a long term storage
	QuestDB	https://www.questdb.org/	https://github.com/bluestreak	Custom	Java			Yes?	No?	
	Riak TS	http://basho.com/posts/business/riak-ts-1-3-r	https://github.com/basho/riak	Riak	Erlang			Yes	Yes, CrossDC - (	Same source tree as Riak itself, different branch. Seems to be not very actively developed. Have some strange limitations on queries (e.x. database metadata defines maximum timerange for select queries)
	Roshi	https://github.com/soundcloud/roshi	https://github.com/soundcloud	Custom	Golang			?	Yes	Time-series Event Storage
	RRDTool	http://oss.oetiker.ch/rrdtool/	http://oss.oetiker.ch/rrdtool/de	RRD	C			No	No	Predecessor of Whisper
	Scylla	http://www.scylladb.com/technology/archited	https://github.com/scylladb/sy	Custom (Inspired	C++			?	Yes	General-Purpose NoSQL database. Cassandra-compatible. Might require a lot of work to actually be suitable as TSDB
	SiriDB	http://siridb.net/	https://github.com/transceptio	Custom	C			?	Yes	Docs looks very nice, very young, suspicious historical reasons
	Thanos	https://github.com/thanos-io/thanos	https://github.com/improbable	GCS + Custom	Golang					Long-term storage for prometheus. Uses Google Compute Storage.
	TimescaleDB	http://www.timescale.com/	https://github.com/timescale/t	PostgreSQL	C			Yes?	Not yet (In TODO)	
	Warp10	http://www.warp10.io/	https://github.com/nicbezen/da	LevelDB, HBase	Java	9 [32]	9 [33]	Yes?	HBase-based	Geo time series data (whatever that means)
	Whisper (Carbon)	http://graphiteapp.org/	https://github.com/graphite-pr	Whisper	Python	12 [34]	12 [35]	Yes [36]	Yes* [37]	
	Whisper (Go-carbon)	https://github.com/lomik/go-carbon	https://github.com/lomik/go-c	Whisper	Golang	12 [38]	12 [39]	Partial [40]	Yes* [41]	Reimplementation of carbon in Go
	CrateDB	https://crate.io	https://github.com/crate/crate	Lucene [42]	Java				Yes	
	VictoriaMetrics	https://victoriametrics.com/	https://github.com/VictoriaMe	custom	golang	<1* [43]	16	Yes	Yes* [44]	
	QuestDB	https://questdb.io/	https://github.com/questdb/qu	custom	C++				?	
	LinDB	https://lindb.io/	https://github.com/lindb/lindb	custom?	golang			Yes	Yes	Early development stages
	arcticDB	https://www.polarsignals.com/	https://github.com/polarsignal	Apache Parquet	golang			Yes	Yes	https://www.polarsignals.com/blog/posts/2022/05/04/introducing-arcticdb/
	TDengine	https://tdengine.com/	https://github.com/taosdata/T	custom	C			Yes	Yes	
	eyros	https://github.com/jpmmapx/eyros	https://github.com/jpmmapx/eyros							
	siridb	http://siridb.net/	https://github.com/SiriDB/siridb-server							
	warp10	https://warp10.io/	https://github.com/senx/warp10-platform							
										Abandoned since 2019

Info [45]	**Name**	**Homepage**	**Source Code**	**Storage Engine**	**Language**	**Bytes/point (ach**	**Bytes/point (wo**	**Dimentions/Tag**	**Clustering**	**Comments** NOTE: YOU CAN COMMENT (ADD SUGGESTIONS) THIS SHEET. For more information about the data - see the bottom of the sheet
	Akumuli	http://akumuli.org	https://github.com	Custom	C++	0.1 [48]	8.5 [49]	Yes	No	**No commits for a half year. Might be dead**
	Argus	https://github.com	https://github.com	Custom	Java			Yes	No?	**DEAD (Archived on github)**
	beringei	https://github.com	https://github.com	Custom	C++			?	No	**DEAD.** Reference implementation of Gorilla whitepaper. In-memory
	Blueflood	http://blueflood.io	https://github.com	Cassandra	Java	8 [50]	8 [51]	Yes	Yes?	**DEAD.**
	BTrDB	https://blog.acolyt	https://github.com	Custom	Golang			No?	Yes?	**Likely dead. Several years without commits. Comercial version's website doesn't work properly.** Academical DB. Papers sounds very nice. Developed for IoT applications
	Catena	https://github.com	https://github.com	Custom	Golang					**DEAD. SHOULD BE USED ONLY FOR HISTORICAL REASONS.**
	Ceres	http://graphiteapp	https://github.com	Ceres	Python	8 [52]	8 [53]	Yes	Yes* [54]	**Abandonware**
	Chronix	http://chronix.io	https://github.com	Lucene	Java	3 [55]	3 [56]	Yes	Yes	**DEAD**
	Cube	https://square.gith	https://github.com	MongoDB	NodeJS			?		**DEAD FOR OVER 4 YEARS. SHOULD BE USED ONLY FOR HISTORICAL REASONS**
	Cyanite	http://cyanite.io/	https://github.com	Cassandra	Clojure			No	Cassandra-base	**DEAD**
	Dalmatiner	https://dalmatiner	https://gitlab.com	Riak	Erlang			Yes [57]	Yes, CrossDC - C	**DEAD.** Docs are very bad. No recomendations on clustering, just mentions that it supports it. One man's project. Sounds very good on paper
	EventQL	https://eventql.io/	https://github.com	Custom	C++			?	Yes	**DEAD.** BETA QUALITY. General Purpose Column-based analytics db, event-based, needs investigation if it's suitable for time-series data
	hawkular	http://www.hawk	https://github.com	Cassandra	Java	12	12	Yes	Cassandra-base	**DEAD.**
	Kenshin	https://github.com	https://github.com	Custom	Python			No	No	**DEAD.** Design goals - make whisper less I/O hungry
	Sidewinder	http://sidewinder.scotya.com/	https://github.com	Custom	Java	3 [58]	4.5 [59]		Yes	Custom DB with own storage, protocol and API. Accepts InfluxDB proto also, there're Collectd and Grafana integrations.
	Tgres	https://github.com	https://github.com	PostgreSQL	Golang	8,11 [60]	8,11 [61]	?	Yes [62]	DEAD
	Timely	https://nationalse	https://github.com	Apache Accumul	Java			?	Yes?	No Updates for half a year. Seems to be dead. Created by NSA
	TrailDB	http://traildb.io/	https://github.com	Custom	C			?	???	General Purpose Event-based database, needs investigation if can be suitable for time-series data
	TritanDB	http://www.tritand	https://github.com	Custom	Kotlin					Claims to be optimized for IoT Workloads
	Vaultaire	https://github.com	https://github.com	Custom	Haskell				Ceph-based	**DEAD FOR OVER 3 YEARS. SHOULD BE USED ONLY FOR HISTORICAL REASONS. Relies on CEPH for data storage**
	vector	https://github.com	https://github.com	Custom	NodeJS			?	No	**DEAD.** Per-host monitoring/collection framework. Suitable for a single-host out of the box, needs to be combined with something else to be useful on scale
	Vulcan	https://github.com	https://github.com	Cassandra	Golang			Yes	Yes?	**Officialy Dead.** Prometheus-compatible long term storage

Output [63]	Diamond [64]	Telegraf [65]	Collectd [66]	Snap [67]	Netdata [68]					
amon		V								
ampq		V								
cassandra				V						
cloudwatch	V	V								
datadog	V	V								
elasticsearch		V		V						
etcd				V						
graphite	V	V	V	V	V					
grafana				V [69]						
graylog		V								
hana				V						
hawkular				V						
heapster				V						
heka				V						
hostedgraphite	V									
http	V		V							
influxdb	V	V		V	V					
kafka		V	V	V	V					
kinesis		V								
kairosdb				V	V					
librato		V								
mongodb			V							
mqtt	V	V								
mysql	V			V						
nats		V								
nsq		V								
opentsdb		V		V	V					
prometheus		V	V		V					
rabbitmq	V			V						
riemann	V	V	V	V						
redis			V							
rrdtool	V									
sentry	V									
sensu			V							
statsd	V									
Name	Source Code	Language	License	Supported Outputs	Extra					
Diamond	https://github.com/python-diamon	Python	MIT	Graphite, InfluxDB, mqtt, m	Outputs and collectors are python scripts					
Telegraf	https://github.com/influxdata/teleg	Go	MIT	amon, ampq, cloudwatch, d	Plugins are compile-time					
Collectd	https://github.com/collectd/collectd	C	MIT	graphite, http, kafka, mongo	Plugins are .so files					
Snap	https://github.com/intelsdi-x/snap	Go	Apache 2.0	influxdb, graphite, opentsdb	All plugins (collectors, outputs) are separate pieces of software that talks over gRPC					
Netdata	https://github.com/firehol/netdata	C	GPLv3	graphite, kairosdb, influxdb,	Can work as a standalone monitoring+alerting+dashboard system. Have Alerting capatibilities.					

	Website	Source Code		Language of Reference Implementation																
Aerospike	https://www.aero	https://github.com	Custom	C	64	64	No	Yes, CrossDC - (	General Purpose NoSQL database, can be used for time-series, but seems to not be a good fit. Bytes/point is an estimation from not very reliable source. Needs reevaluating											
Tarantool	https://tarantool.c	https://github.com	Custom	C	N/A**** [71]	N/A**** [72]	N/A****	Yes	More application framework with a database, can be used for Time-Series, but requires A LOT of work for that											
Cassandra				Java																
LevelDB	https://github.com	https://github.com	Custom	C++/Go	?	?	?	?												
RocksDB				C++																
Clickhouse				C++																
ScyllaDB				C++																

This list will contain just names of databases I think would be iteresting to test:					
Requirements:					
Tags (Metrics 2.0 or InfluxDB Line Protocol)					TimescaleDB
Reliability					clickhouse
Retentions					m3db
					eventql
					siridb
					Roshi
					DalmatinerDB
					Sidewinder

[1] Self-link: https://goo.gl/BRqzdG

This sheet is an attempt to structure basic information about time-series databases (or stuff that can be used as them to some extent).

At this moment I'm against providing performance evaluation of those databases, because it's very hard to create a fair environment, and if it's not fair - test will be useless.

Useful link with some thoughts on TSDBs: https://misfra.me/2016/04/09/tsdb-list/ (updated once in a while)

I'll only add an OpenSource database (at least basic functionality must be opensource under any OSI approved license)

[2] average for large metrics. If database have compression - better to provide either values (range preferrably) and links to the experiments (if any), otherwise - should be a value from whitepaper/docs and if there are any notes - they should be added on a "Comments" field.

[3] For tags support I'll place Yes if and only if there is a history of changes there, e.x. query by dimentions will return the real state of things, not the current one.

[4] https://docs.google.com/presentation/d/17opE2U2aIe1TFYJgr0Z8Dd2fJhy2x0YbReCVeCz6uhA/edit?usp=sharing

FOSDEM 2018 talk

[5] Tags are not supported at the moment, as per https://github.com/criteo/biggraphite/blob/master/biggraphite/plugins/tags.py

[6] It relies on PostgreSQL to store data, without any custom compression, etc

[7] It relies on PostgreSQL to store data, without any custom compression, etc

[8] Custom (not Postgres-Based)
Highly depends on data model (thx. to dzhdanov)

[9] Depends on Clickhouse version and config.

If zero-timestamp is used and double delta compression on Clickhouse side - could be lower than 2.

2 is however average for optimal settings according to the Author (lomik).

According to my own tests with clickhouse and default compression, with "zero-timestamp" turned off it gave:
5.4 - based on autogenerated data
6.5- based on real-world data set

Depends on a dataset even with generic compression algorithm it can go down to about 2.5 b/point (for system metrics for example)

[10] According to my own tests
5.4 - based on autogenerated data
6.5- based on real-world data set

[11] https://github.com/lomik/graphite-carbon only supports current state of dimentions.

Though it's possible to implement history of changes

[12] May be a bit better than graphite-clickhouse, but likely not significant as both stores same data in a mostly same way

[13] May be a bit better than graphite-clickhouse, but likely not significant

[14] Accroding to https://www.elastic.co/blog/elasticsearch-as-a-time-series-data-store - post optimize 508M for 23M points, 508/23 ~= 22
Valid for 2.0, might be better in 5.x

[15] Accroding to https://www.elastic.co/blog/elasticsearch-as-a-time-series-data-store - post optimize 2200M for 23M points = 2200/23 ~ 96
Valid for 2.0, might be better in 5.x

[16] https://julien.danjou.info/talks/storing-metrics-at-scale-with-gnocchi.pdf
6.25 - avg

[17] https://julien.danjou.info/talks/storing-metrics-at-scale-with-gnocchi.pdf
6.25 - avg

[18] 2 bytes/point according to official response: https://community.influxdata.com/t/how-much-disk-does-influxdb-consume/31
Depends on the data set it can go below 1 byte/point

[19] Acroding to official documentation, depends on randomness of data

[20] https://groups.google.com/forum/#!topic/kairosdb-group/B77fsHCyFtk - according to some random tests.

On a good data it can be 4 bytes per point, on worse data can be worse

[21] https://groups.google.com/forum/#!topic/kairosdb-group/B77fsHCyFtk - according to some random tests.

On a good data it can be 4 bytes per point, on worse data can be worse

[22] according to one of the authors.

[23] according to one of the authors.

[24] According to one of the authors

[25] According to one of the authors

[26] Can store them but current version don't have Documented API to query them

[27] Cassandra as storage is ok

Metrictank's own clustering is very basic and relies on kafka partitions (or graphite relays), HA is also possible, but master promotions are manualv

[28] http://opentsdb.net/faq.html

12 bytes with compression, without HDFS replication.

39 - minimal amount without compression, + 6b/tag

[29] http://opentsdb.net/faq.html

12 bytes with compression, without HDFS replication.

39 - minimal amount without compression, + 6b/tag

[30] Depends on compression algorithm and data
https://prometheus.io/docs/operating/storage/

[31] Depends on compression algorithm and data
https://prometheus.io/docs/operating/storage/

[32] Uncertain, not from docs

[33] Uncertain, not from docs

[34] Excluding metadata overhead

[35] Excluding metadata overhead

[36] TagDB in current master (what will become 1.1.0).

[37] there are several projects that provides clustering for that kind of metrics - github.com/go-graphite/ for whisper-based, graphtie-web can also do some sort clustering

[38] Excluding metadata overhead

[39] Excluding metadata overhead

[40] Only current state with https://github.com/kanatohodets/carbonsearch

Requires separate daemon, query language is rudimental

[41] there are several projects that provides clustering for that kind of metrics - github.com/go-graphite/ for whisper-based, graphtie-web can also do some sort clustering

[42] ElasticSearch to be precise

[43] Main article about Compression: https://www.google.com/url?q=https://medium.com/@valyala/victoriametrics-achieving-better-compression-for-time-series-data-than-gorilla-317bc1f95932&sa=D&ust=1559117797555000&usg=AFQjCNELWLSFF1ETBmLTqEOeDLOsWFiKwg

According to:
https://medium.com/@valyala/high-cardinality-tsdb-benchmarks-victoriametrics-vs-timescaledb-vs-influxdb-13e6ee64dd6b
and
https://medium.com/@valyala/measuring-vertical-scalability-for-time-series-databases-in-google-cloud-92550d78d8ae

[44] Cluster version is separate from single node one: https://github.com/VictoriaMetrics/VictoriaMetrics/blob/cluster/README.md

[45] Self-link: https://goo.gl/BRqzdG

This sheet is an attempt to structure basic information about time-series databases (or stuff that can be used as them to some extent).

At this moment I'm against providing performance evaluation of those databases, because it's very hard to create a fair environment, and if it's not fair - test will be useless.

Useful link with some thoughts on TSDBs: https://misfra.me/2016/04/09/tsdb-list/ (updated once in a while)

I'll only add an OpenSource database (at least basic functionality must be opensource under any OSI approved license)

[46] average for large metrics. If database have compression - better to provide either values (range preferrably) and links to the experiments (if any), otherwise - should be a value from whitepaper/docs and if there are any notes - they should be added on a "Comments" field.

[47] For tags support I'll place Yes if and only if there is a history of changes there, e.x. query by dimentions will return the real state of things, not the current one.

[48] Depends on dataset, can be worse than 4 bytes on dataset with random floats

[49] Depends on dataset, can be worse than 4 bytes on dataset with random floats

[50] https://github.com/rackerlabs/blueflood/wiki/FAQ

[51] https://github.com/rackerlabs/blueflood/wiki/FAQ

[52] Excluding constant metadata overhead

[53] Excluding constant metadata overhead

[54] there are several projects that provides clustering for that kind of metrics - github.com/go-graphite/ for whisper-based, graphtie-web can also do some sort clustering

[55] It's UNCLEAR what's the characteristics, because they doesn't specify that in bytes/point. On https://www.usenix.org/system/files/conference/fast17/fast17-lautenschlager.pdf they claim to be 20% better than InfluxDB at that point, that was 4 b/point

[56] It's UNCLEAR what's the characteristics, because they doesn't specify that in bytes/point. On https://www.usenix.org/system/files/conference/fast17/fast17-lautenschlager.pdf they claim to be 20% better than InfluxDB at that point, that was 4 b/point

[57] Requires external PostgreSQL for tags.

[58] http://sidewinder.srotya.com/docs/#/designs/compression

[59] http://sidewinder.srotya.com/docs/#/designs/compression

[60] Random source:
https://news.ycombinator.com/item?id=13247598

[61] Random source:
https://news.ycombinator.com/item?id=13247598

[62] Custom, not postgres based

[63] This table is based on what software says about it's Outputs. There was no sanity checks on that done by author - e.x. no checks were made to verify if there is any use case for having data in Hana for example

[64] Language: Python
License: MIT

Outputs and collectors are python scripts

[65] Language: Go
License: MIT

Plugins are compile-time

[66] Language: C
License: MIT

Plugins are .so librarires

[67] Language: Go
License: Apache 2.0

All plugins (collector and outputs) are separate pieces that talks over gRPC

[68] Language: C
License: GPLv3

Can work as a standalone system
Have Alerting capatibilities.

[69] Direct output to Grafana (https://github.com/raintank/snap-app) - it doesn't store data for very long, but allows you to use Grafana to get current state of metrics

See https://grafana.com/blog/2016/03/31/using-grafana-with-intels-snap-for-ad-hoc-metric-exploration/ for more details

[70] Aerospike haves community edition money-free (sends telemetry) and two kinds of commercial. All three supports HA for needed nodes number and replication factor. HA is the main feature. Therefore they opened HA + Clustering for all editions.

[71] As it's a framework - it's not strictly defined by the database.

[72] As it's a framework - it's not strictly defined by the database.