Welcome to Apache HBase™

Apache (https://www.apache.org/) HBase™ is the Hadoop (https://hadoop.apache.org/) database, a distributed, scalable, big data store.

Use Apache HBase™ when you need random, realtime read/write access to your Big Data. This project's goal is the hosting of very large tables -- billions of rows X millions of columns -- atop clusters of commodity hardware. Apache HBase is an open-source, distributed, versioned, non-relational database modeled after Google's Bigtable: A Distributed Storage System for Structured Data (https://research.google.com/archive/bigtable.html) by Chang et al. Just as Bigtable leverages the distributed data storage provided by the Google File System, Apache HBase provides Bigtable-like capabilities on top of Hadoop and HDFS.

Download

Click **here (downloads.html)** to download Apache HBase™.

Features

- Linear and modular scalability.
- Strictly consistent reads and writes.
- Automatic and configurable sharding of tables
- Automatic failover support between RegionServers.
- Convenient base classes for backing Hadoop MapReduce jobs with Apache HBase tables.
- Easy to use Java API for client access.
- Block cache and Bloom Filters for real-time queries.
- Query predicate push down via server side Filters
- Thrift gateway and a REST-ful Web service that supports XML, Protobuf, and binary data encoding options
- Extensible jruby-based (JIRB) shell
- Support for exporting metrics via the Hadoop metrics subsystem to files or Ganglia; or via JMX

More Info

See the Architecture Overview (book.html#arch.overview), the Apache HBase Reference Guide FAQ (book.html#faq), and the other documentation links.

Export Control
The HBase distribution includes cryptographic software. See the export control notice here (export_control.html)

Code Of Conduct
We expect participants in discussions on the HBase project mailing lists, Slack and IRC channels, and JIRA issues to abide by the Apache Software Foundation's Code of Conduct (https://apache.org/foundation/policies/conduct.html) . More information can be found here (coc.html).

License
Apache HBase is licensed under the Apache License, Version 2.0 (https://www.apache.org/licenses/LICENSE-2.0)

Trademarks
Apache HBase, HBase, Apache, the Apache feather logo, and the Apache HBase project logo are either registered trademarks or trademarks (https://www.apache.org/foundation/marks/list/) of The Apache Software Foundation in the United States and other countries.

Thanks
Thanks for all the sponsors, who are supporting Apache (https://www.apache.org/foundation/thanks.html) or supporting the HBase project (sponsors.html)!

Security and Vulnerability information
See the Security (book.html#security) chapter in the Apache HBase Reference Guide (book.html#faq), and the general Apache Security information (https://apache.org/security/) !

News

July 20th, 2019 HBaseCon, Asia 2019 (https://easychair.org/cfp/hbaseconasia-2019) Beijing, China

May 21st, 2019 NoSQL Day 2019 (https://dataworkssummit.com/nosql-day-2019/) Washington DC.

August 17th, 2018 HBaseCon Asia 2018 (https://hbase.apache.org/hbaseconasia-2018/) @ Gehua New Century Hotel, Beijing, China. CFP open, see site for details!

June 18th, 2018 HBaseCon North America West 2018 (https://hbase.apache.org/hbasecon-2018) @ San Jose Convention Center, San Jose, CA, USA. registration still open, see site for details!

August 4th, 2017 HBaseCon Asia 2017 (https://easychair.org/cfp/HBaseConAsia2017) @ the Huawei Campus in Shenzhen, China

Old News (old_news.html)

The next Apache Event:

(https://www.apache.org/events/current-event.html)

Last Published: 2024-03-21

NoSQL?

HBase is a type of "NoSQL" database. "NoSQL" is a general term meaning that the database isn't an RDBMS which supports SQL as its primary access language, but there are many types of NoSQL databases: BerkeleyDB is an example of a local NoSQL database, whereas HBase is very much a distributed database. Technically speaking, HBase is really more a "Data Store" than "Data Base" because it lacks many of the features you find in an RDBMS, such as typed columns, secondary indexes, triggers, and advanced query languages, etc.

However, HBase has many features which supports both linear and modular scaling. HBase clusters expand by adding RegionServers that are hosted on commodity class servers. If a cluster expands from 10 to 20 RegionServers, for example, it doubles both in terms of storage and as well as processing capacity. An RDBMS can scale well, but only up to a point - specifically, the size of a single database server - and for the best performance requires specialized hardware and storage devices. HBase features of note are:

- Strongly consistent reads/writes: HBase is not an "eventually consistent" DataStore. This makes it very suitable for tasks such as high-speed counter aggregation.
- Automatic sharding: HBase tables are distributed on the cluster via regions, and regions are automatically split and re-distributed as your data grows.
- Automatic RegionServer failover
- Hadoop/HDFS Integration: HBase supports HDFS out of the box as its distributed file system.
- MapReduce: HBase supports massively parallelized processing via MapReduce for using HBase as both source and sink.
- Java Client API: HBase supports an easy to use Java API for programmatic access.
- Thrift/REST API: HBase also supports Thrift and REST for non-Java front-ends.
- Block Cache and Bloom Filters: HBase supports a Block Cache and Bloom Filters for high volume query optimization.

- Operational Management: HBase provides build-in web-pages for operational insight as well as JMX metrics.

When Should I Use HBase?

HBase isn't suitable for every problem.

First, make sure you have enough data. If you have hundreds of millions or billions of rows, then HBase is a good candidate. If you only have a few thousand/million rows, then using a traditional RDBMS might be a better choice due to the fact that all of your data might wind up on a single node (or two) and the rest of the cluster may be sitting idle.

Second, make sure you can live without all the extra features that an RDBMS provides (e.g., typed columns, secondary indexes, transactions, advanced query languages, etc.) An application built against an RDBMS cannot be "ported" to HBase by simply changing a JDBC driver, for example. Consider moving from an RDBMS to HBase as a complete redesign as opposed to a port.

Third, make sure you have enough hardware. Even HDFS doesn't do well with anything less than 5 DataNodes (due to things such as HDFS block replication which has a default of 3), plus a NameNode.

HBase can run quite well stand-alone on a laptop - but this should be considered a development configuration only.

What Is The Difference Between HBase and Hadoop/HDFS?

HDFS is a distributed file system that is well suited for the storage of large files. Its documentation states that it is not, however, a general purpose file system, and does not provide fast individual record lookups in files. HBase, on the other hand, is built on top of HDFS and provides fast record lookups (and updates) for large tables. This can sometimes be a point of conceptual confusion. HBase internally puts your data in indexed "StoreFiles" that exist on HDFS for high-speed lookups.