About Apache HAWQ®

Apache HAWQ® is a Hadoop native SQL query engine that combines the key technological advantages of MPP database with the scalability and convenience of Hadoop. HAWQ reads data from and writes data to HDFS natively. HAWQ delivers industry-leading performance and linear scalability. It provides users the tools to confidently and successfully interact with petabyte range data sets. HAWQ provides users with a complete, standards compliant SQL interface. More specifically, HAWQ has the following features:

- On-premise or cloud deployment
- Robust ANSI SQL compliance: SQL-92, SQL-99, SQL-2003, OLAP extension
- Extremely high performance. many times faster than other Hadoop SQL engines.
- World-class parallel optimizer
- Full transaction capability and consistency guarantee: ACID
- Dynamic data flow engine through high speed UDP based interconnect
- Elastic execution engine based on on-demand virtual segments & data locality
- Support multiple level partitioning and List/Range based partitioned tables.
- Multiple compression method support: snappy, gzip, quicklz, RLE
- Multi-language user defined function support: python, perl, java, c/c++, R
- Advanced machine learning and data mining functionalities through MADLib
- Dynamic node expansion: in seconds
- Most advanced three level resource management: Integrate with YARN and hierarchical resource queues.
- Easy access of all HDFS data and external system data (for example, HBase)
- Hadoop Native: from storage (HDFS), resource management (YARN) to deployment (Ambari).
- Authentication & Granular authorization: Kerberos, SSL and role based access
- Advanced C/C++ access library to HDFS and YARN: libhdfs3 & libYARN
- Support most third party tools: Tableau, SAS et al.
- Standard connectivity: JDBC/ODBC

Publications

[1] Lei Chang et al: HAWQ: a massively parallel processing SQL engine in hadoop. SIGMOD Conference 2014: 1223-1234

[2] Mohamed A. Soliman et al: Orca: a modular query optimizer architecture for big data. SIGMOD Conference 2014: 337-348

[3] Lyublena Antova et al: Optimizing queries over partitioned tables in MPP systems. SIGMOD Conference 2014: 373-384

[4] Amr El-Helw et al: Optimization of Common Table Expressions in MPP Database Systems. PVLDB 8(12): 1704-1715 (2015)