REPORT

Splice Machine: SQL-on-Hadoop® Evaluation Guide

splice MACHINE

www.splicemachine.com

Table of Contents

Hadoop: Moving Beyond the Big Data Hype

Let's face it. There is a lot of hype surrounding Big Data and Hadoop, the de facto Big Data technology platform. Companies want to mine and act on massive data sets, or Big Data, to unlock insights that can help them improve operational efficiency, delight customers, and leapfrog their competition.

Hadoop has become popular to store massive data sets because it can distribute them across inexpensive commodity servers. Hadoop is fundamentally a file system (HDFS or Hadoop Distributed File System) with a specialized programming model (MapReduce) to process the data in the files.

Big Data has not lived up to expectations so far, partly because of **limitations of Hadoop as a technology:**

- **A file system, not a database** – Even with MapReduce, Hadoop does not provide easy access to individual records or record sets that are a small subset of the total data. It lacks many of the capabilities that a typical database has to organize and ensure data consistency.
- **Designed for batch analysis** – To find even one record, MapReduce must scan all records. Thus, it is really designed for a large, batch analysis that aggregates or processes most, if not all, of a massive data set. As such, Hadoop cannot support the interactive, ad-hoc queries required for many applications.
- **No updates** – As an immutable file system, no data can be updated or deleted in Hadoop. So if you have 100 TB of data, all of it must be rewritten (like through a daily ETL process) if any of it changes. This can be a big issue for many companies. If there is any constant in business, it's that everything changes, and both the data and analysis need to stay up-to-date to remain relevant.
- **Requires Java programs to manage data** – Hadoop requires complex, specialized MapReduce programs, typically written in Java, to manage its data. However, Java programmers, let alone ones trained in MapReduce, are rare at most companies. This often becomes a huge bottleneck, as every data request from a data scientist or business analyst must go through a Java programmer.

As a result of these limitations, Hadoop has become the "roach motel" of data for many companies – easy to get data in, hard to get it out.

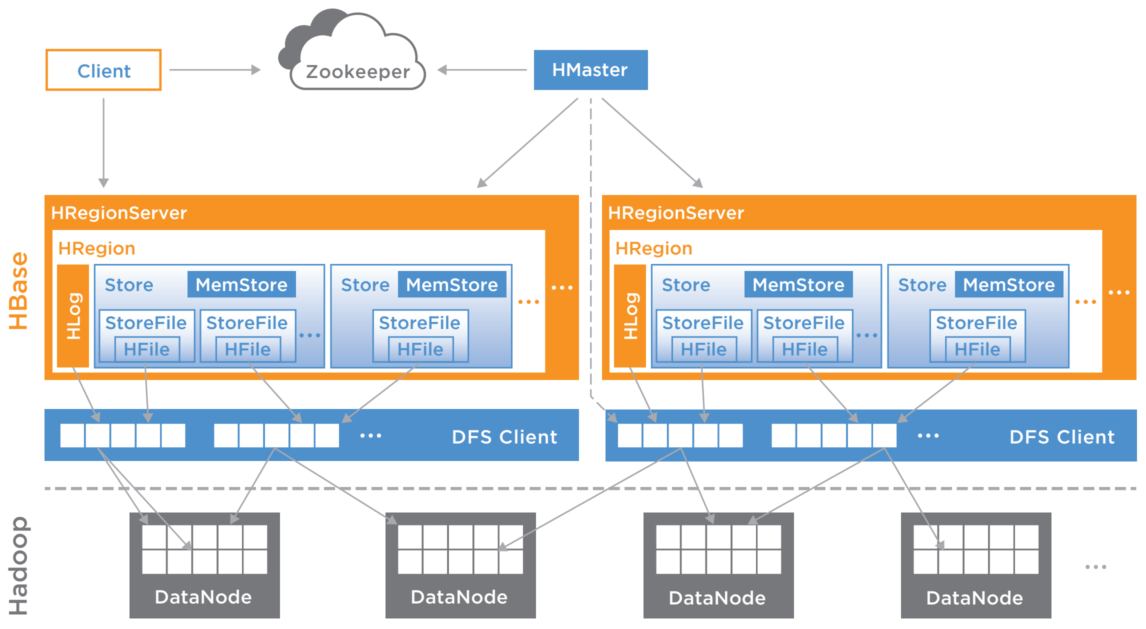
HBase: The Database on Hadoop

Built on top of HDFS, HBase is the database on Hadoop. It is a simple key-value store that supports a limited number of operations: Get, Put, Scan, and Delete. It provides fast lookups and updates, but HBase still has some major limitations:

- **No SQL** – HBase requires Java code to input or extract data. As mentioned, Java programmers are rare at most companies and can quickly become a bottleneck for new data requests.
- **No Transactions** – Transactions are critical to enable real-time updates across multiple rows and tables. Without transactions, updates may lead to data loss or corruption.
- **No Joins** – Hbase does not provide the ability to join data across multiple tables. This means you have to write Java code to coordinate queries across tables and to do analytics on multiple dimensions.
- **No Secondary Indexes** – Hbase does not support secondary indexes, which makes non-primary key lookups slower table scans.

Given these limitations, HBase has been used mainly for applications needing simple, fast lookups on single rows in a table. HBase is rarely used for analytics or applications that require cross-row or cross-table operations.

fig. 1
HBase Architecture



source: http://www.larsgeorge.com/2009/10/hbase-architecture-101-storage.html

Struggling with Existing Databases

Meanwhile the same explosion of web, social, mobile and machine-generated data driving the Big Data trend has also started to overwhelm traditional shared-disk databases (RDMBSs).

MySQL, PostgreSQL, and SQL Server databases start to encounter scaling issues at below a terabyte of data without manual sharding. However, manual sharding is not a viable option for most companies as it requires a partial rewrite of every application. It also becomes a maintenance nightmare to periodically rebalance shards as they grow too large.

Oracle and IBM DB2 databases can scale up beyond a few terabytes using expensive, specialized hardware. With costs that can exceed millions per server, scaling quickly becomes cost-prohibitive for most companies.

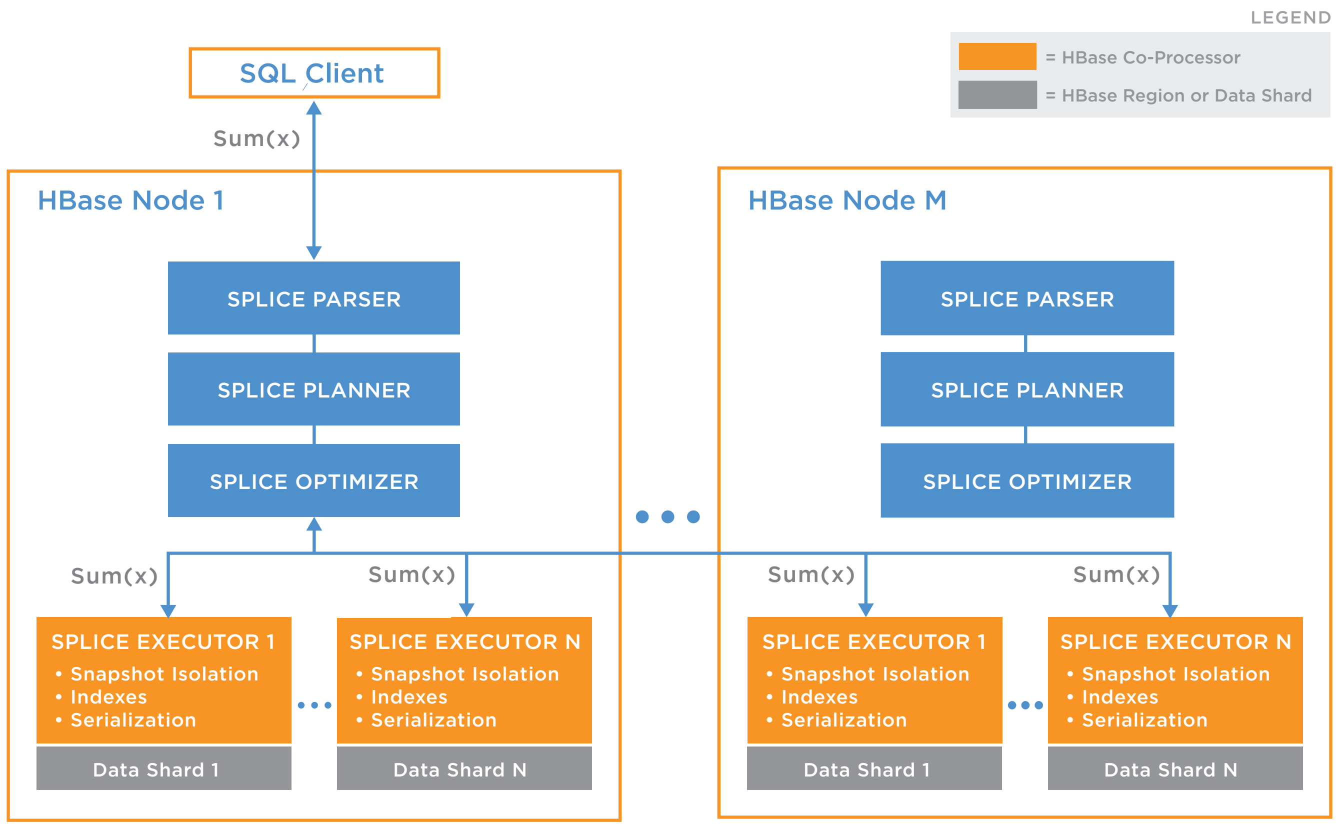
Solution: SQL-on-Hadoop

SQL-on-Hadoop technologies include a SQL layer or even a full SQL database over Hadoop. SQL-on-Hadoop solutions have become very popular recently as they solve the data management issues of Hadoop and provide a scale-out alternative for traditional RDBMSs.

For existing Hadoop users, the ubiquity of SQL dramatically improves data management. Almost all data and business analysts already know and use SQL to manage data by themselves in current systems. Thousands of existing SQL tools and Business Intelligence (BI) can now connect to Hadoop data through a standard ODBC connection. There also millions of custom SQL applications that can now update and act on that data.

For existing databases experiencing scaling issues, SQL-on-Hadoop solutions can provide a full SQL database that can scale out on commodity hardware. With standard SQL, it can eliminate application rewrites to access scale-out technology. With scalability proven into petabytes on inexpensive servers for Hadoop, SQL-on-Hadoop also provides a highly scalable data platform that does not require expensive, specialized hardware.

fig. 2
The Distributed Computing Architecture of Splice Machine is an Example of How SQL-on-Hadoop Solutions Scale Out



Reasons to Use SQL-on-Hadoop Solutions

To summarize, we present seven reasons to use SQL-on-Hadoop, depending on your current goals:

FOR PEOPLE ALREADY USING HADOOP

1. **Use existing SQL trained personnel** – no retraining required to manage Hadoop-based data
2. **Use existing SQL tools, BI tools** – be productive immediately with existing SQL-based tools
3. **Use existing SQL apps** – no rewrites required to manage Hadoop-based data

FOR USERS OF EXISTING DATABASES

4. **MySQL, PostgreSQL, or SQL Server can't scale** – scale out without the heavy coding required to manually shard
5. **Oracle or IBM DB2 too expensive** – scale out without spending millions per server

FOR NEW APPLICATIONS

6. **"Future proof "database platform** – leverage a database platform that can handle your future data growth and prevent a database migration in the future
7. **Real-time, data-driven applications** – enable a new breed of real-time, analytical applications that collect, analyze, AND react to massive amounts of data in real-time to delight customers and operate complex systems

Case Studies

Here are some case studies on how leading companies are using SQL-on-Hadoop to solve real business problems:

CASE STUDY: OPERATIONAL ANALYTICS

Company: Over $20B Retailer

Application: Operational Analytics

Current Infrastructure: MySQL databases

Challenges:

- Not scaling
- Poor query performance
- Can't update data fast enough

Solution: Splice Machine replaces MySQL database

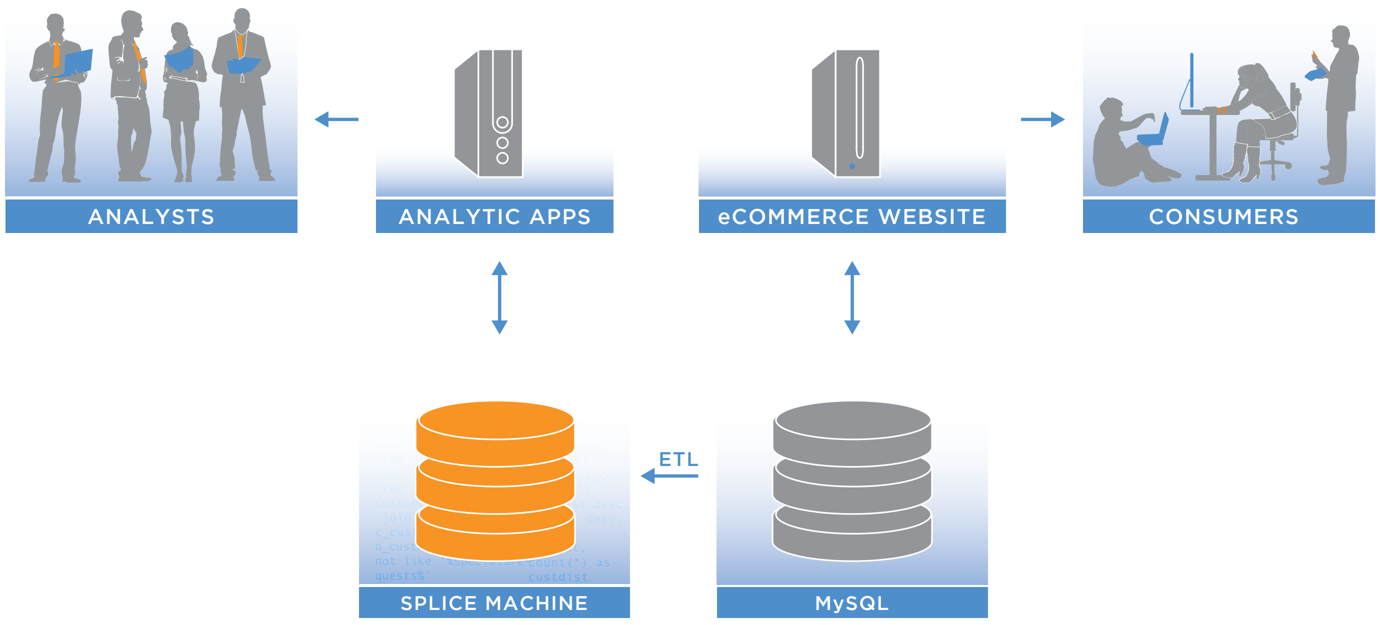
Benefits:

- Linear scaling
- Parallelized queries

Results:

- Near real-time updates
- Excellent query performance

fig. 3
SQL-on-Hadoop databases can deliver scalability and performance that MySQL can't achieve while not disrupting existing workflows.



CASE STUDY: CONSUMER MARKETING

Company: $1B Marketing Services Company

Application: Real-Time Campaign Management

Current Infrastructure: Unica, Cognos, Oracle databases

Challenges:

- Oracle too expensive
- Poor query performance

Solution: Splice Machine replaces Oracle database

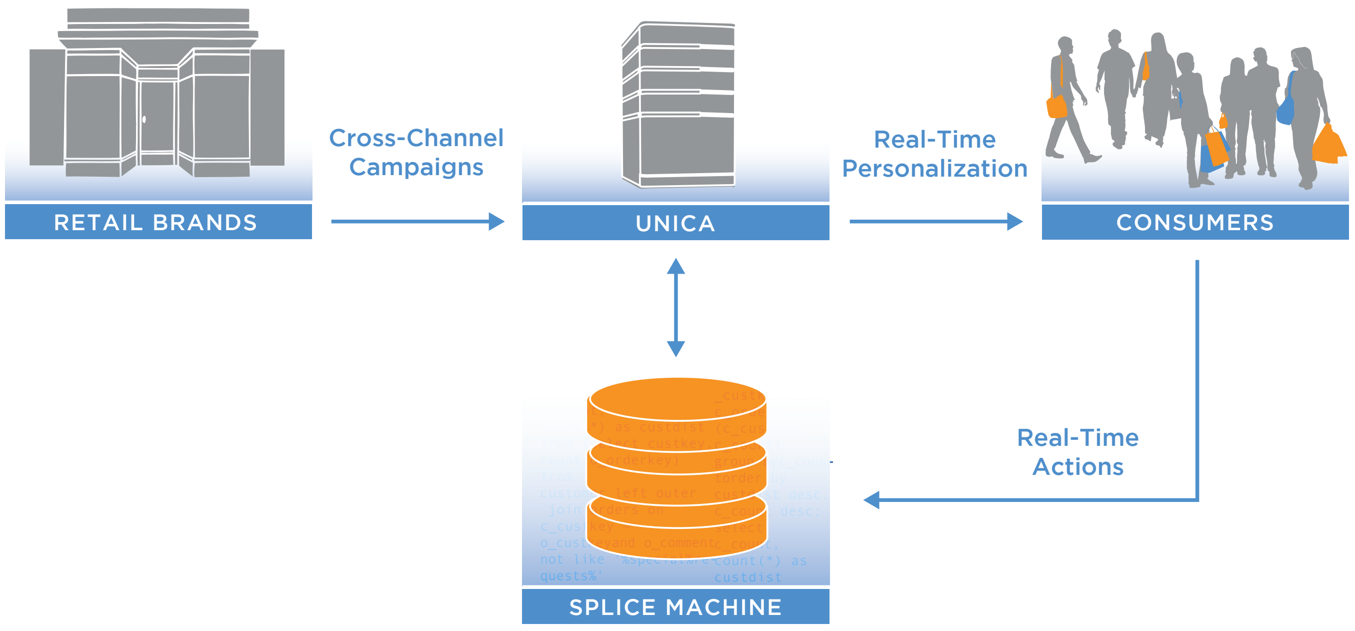
Benefits:

- Real-time updates with transactional integrity
- Scale out with commodity hardware

Results:

- 10x improvement in price/performance over Oracle databases
- Improved query performance by 30-50%

fig. 4
A SQL-on-Hadoop database like Splice Machine is able to replace an existing Oracle database because of ANSI SQL support and simple integration with existing Unica and Cognos databases.



The SQL-on-Hadoop Checklist

If you have read this far, you are probably considering a SQL-on-Hadoop solution. However with the many options available, this can be a daunting task. Here are some questions to ask to help ensure that you get a SQL-on-Hadoop solution that meets your needs:

REAL-TIME

1. **Can it support real-time apps?** This includes real-time operational analytics and traditional operational applications such as web, mobile, and social applications as well as enterprise software. Real-time applications require queries to respond in milliseconds to seconds versus minutes to hours.

2. **Is it really real-time?** This means real-time queries on real-time data. Some solutions claim to be real-time because they can do real-time ad-hoc queries, but it is not real-time if the data is from yesterday's ETL (Extract, Transform, Load).

3. **Can it perform reliable, real-time updates?** Database transactions are required to perform real-time updates without data loss or corruption. This is even important in analytics, as data and secondary indices need to be updated together to ensure consistency.

SQL

4. **Is it true SQL?** In other words, it should support standard ANSI SQL. Many solutions provide a limited, unusual variant that requires retraining and partially rewriting applications.

5. **Can it support concurrent user updates consistently without locking up the database?** As mentioned above, transactions are critical to supporting concurrent users with reliable, real-time updates. Transactions need to span multiple rows and tables. They also should be lockless for high throughput and to avoid troublesome deadlock and starvation conditions that freeze up applications and render them useless.

6. **Can you customize with UDFs?** User Defined Functions (UDFs) enable you to add custom functionality to the database. You should be able to write the UDFs in a high-level programming language and parallelize the processing of the UDFs across nodes.

7. **Does it support triggers?** Triggers watch for certain events (e.g., record insert or deletion) to audit changes or to enforce business rules. This is often critical functionality for many applications.

8. **Does the solution support column-based privileges?** Can you specify what columns are accessible to user, groups, and roles?

CONNECTIVITY

9. **Can you connect to existing tools through ODBC/JDBC connectors?** ODBC/JDBC connectors allow you to use existing SQL tools such as DbVisualizer™ and BI tools such as Tableau®.

10. **Is there a RESTful API?** RESTful APIs enable developers to quickly build web-based applications.

EFFICIENCY

11. **Can it efficiently handle sparse data?** In many large data sets, each attribute or column may be sparsely populated. In traditional databases, an empty value must still be stored as a null, which consumes storage. Modern databases should not require nulls for empty values. For instance if a table has 500 columns and a row only needs 2 columns, then the database should only store 2 values and not 498 nulls.

12. **Can you add a column without table scans?** Data requirements change frequently and often require schema changes. However, you don't want to disrupt the business to make those changes. Adding a column should not require full table scans.

13. **Is the data compressed?** Hadoop reduces storage costs, but it's still not free due to replication needed for availability. Any solution should compress the data by at least 5x, preferably 10x.

PERFORMANCE

14. **Can it perform fast lookups on small subset of the data?** Most analysts want to perform interactive, ad-hoc queries on a small subset of data in Hadoop. They don't want to do batch analysis that takes minutes, if not hours, to complete.

15. **Does it support secondary indexes?** Often data is organized along one dimension for fast updating (such as a customer number) but later must be looked up by other dimensions (such as zip code). Secondary indexes enable databases to lookup data across many dimensions efficiently.

16. **Can it deliver fast performance on massive joins?** Analysis on massive data sets often requires at least billion row by billion row joins. No one wants to wait hours for those joins to finish or fail due to memory limitations on nodes.

17. **Does it provide multiple join strategies?** Joins combine data from multiple tables. With a distributed infrastructure like Hadoop that handles very large data sets, multiple join strategies such as nested loop, sort-merge, and broadcast joins are needed to ensure fast join performance.

18. **Is there a cost-based optimizer?** Performance on large data sets greatly depends on choosing the right execution strategy. Simple rules-based optimizers are not enough. Cost-based optimizers that account for the actual cost to execute a query are critical to optimal query performance.

19. **Can data import be parallelized?** Many applications require batch input of data sources which can create a bottleneck. Can the platform use all the available computing resources to import data?

20. **Does the execution of queries exploit data locality?** In a Hadoop-based system, data is distributed across many machines. A high-performance solution must move computation to where the data is stored locally versus always experiencing the network latency to move data across nodes.

We compiled this checklist working with many companies. Choosing a SQL-on-Hadoop solution is a critical decision that will have a long-term impact on your application infrastructure. While you may not need a solution that meets all of these criteria currently, it's important to consider how your requirements may change over time.

Splice Machine: A Real-Time SQL-on-Hadoop Database

Splice Machine provides a real-time SQL-on-Hadoop database designed for Big Data applications. It provides the following features and benefits:

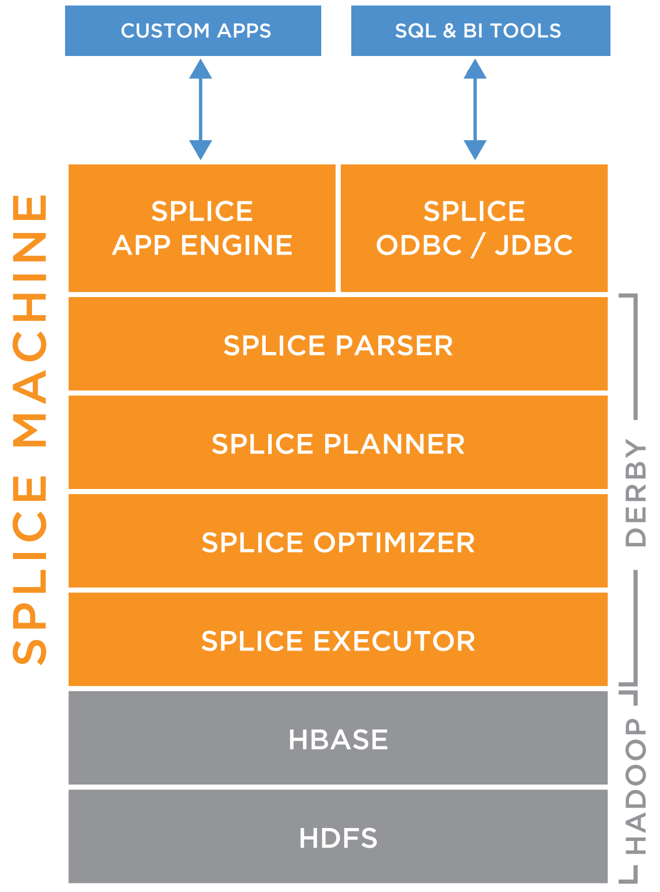
STANDARD ANSI SQL TO LEVERAGE EXISTING SQL TOOLS & SKILL SETS

Splice Machine is an ANSI SQL-compliant database on Hadoop that includes transactions, joins, secondary indexes and UDFs. It also includes ODBC or JDBC drivers for seamless connectivity to BI apps and SQL tools.

COST-EFFECTIVE SCALING WITH COMMODITY HARDWARE

Splice Machine utilizes the proven auto-sharding capability in HBase to provide massive scalability across commodity hardware, even up to dozens of petabytes.

fig. 5
The Splice Machine Solution Stack



REAL-TIME UPDATES WITH TRANSACTIONAL INTEGRITY

Splice Machine provides ACID (Atomicity, Consistency, Isolation, Durability) transactions across multiple rows and tables to enable real-time updates without data loss or corruption. Based on Google Percolator, Splice Machine developed a state-of-the-art, distributed snapshot design that provides high throughput without locking.

HIGH PERFORMANCE, DISTRIBUTED COMPUTING ARCHITECTURE

Splice Machine delivers massive parallelization by placing its parser, planner and optimizer on each node and pushing computation down to each distributed data shard.

FLEXIBLE, GENERAL-PURPOSE DATABASE PLATFORM

Splice Machine can power both OLAP and OLTP workloads, as well as efficiently handle sparse data and non-disruptive schema changes because it uses HBase as its schemaless storage engine.

Learn More

The SQL-on-Hadoop space is constantly evolving.

Please visit **www.splicemachine.com** for up-to-date information or contact us at **info@splicemachine.com**.

splice MACHINE

182 2nd Street, #200
San Francisco, CA 94105
(415) 857-2111
www.splicemachine.com
info@splicemachine.com