CelerData

StarRocks vs. ClickHouse, Apache Druid®, and Trino

Benchmark and Comparison Guide

October 2022

Introduction

Over the last decade, business operations from marketing to product development have become increasingly dependent on the vast stores of data harvested by the enterprise. In fact, it should come as no surprise to anyone that to succeed in the market today, organizations must be more efficient than their competitors when it comes to transforming these huge data volumes into critical business insights.

Delivering that analytics efficiency has been a tall order, however, even for the best engineering teams. Contributing to this challenge is the fact that there is no such thing as the perfect data model. Every type of data has its own model that it works best under, and this reality has forced data engineers to make compromises in query performance, real-time capabilities, maintenance effort, and more just to keep data moving throughout the enterprise.

Modern business analytics demands an OLAP database system capable of delivering excellent performance in both wide-table and multi-table scenarios while also reducing the workloads of data engineers and enabling users to query any dimension of data in real time. This guide will explore the leading available solutions and put them to the test against each other so that enterprises looking to improve their analytics performance can make the right choice based on their needs.

Evaluating the Leading OLAP Database Solutions

Enterprises have no shortage of options when it comes to database solutions. Choice does become limited, however, for businesses working with extreme data volumes, high concurrency, and real-time performance requirements. For analytics teams operating under these constraints, four solutions stand out in their ability to adequately address most, if not all, of the aforementioned criteria. Those solutions are:

StarRocks

A relative newcomer when compared to the other legacy solutions evaluated in this report, StarRocks has quickly become the favored solution by hundreds of enterprises worldwide. While hailed for its ultra-high performance, StarRocks is unique in its ability to eliminate the need for de-normalized tables and reduce engineering overhead in the form of instant updates and deletes as well as easy integrations with popular business intelligence tools.

Apache Druid®

One of the world’s most recognized open-source projects. Apache Druid® boasts instant data visibility, ad-hoc analysis support, and high concurrency particularly when it comes to powering UIs where an interactive user experience is needed.

ClickHouse

Another popular solution, ClickHouse adoption has grown substantially in recent years for delivering database performance that put it in a class of its own. While the competition has caught up in many cases by now, ClickHouse is still the leading choice for many enterprises that are looking for an open solution that can support data-intensive analytics alongside their column-oriented databases.

Trino

Previously known as PrestoSQL, Trino delivers ultra-high query performance for big data analytics across a wide variety of scenarios. Offering query federation, in-place analysis, and the ability to run Trino on-premises or on the cloud, Trino stands out amongst its peers for its versatility.

Query Latency Performance: Wide-Table and Multi-Table

While all the solutions listed above succeed in delivering exceptional performance, it's vital to understand just how they fare under a variety of scenarios. After all, no two enterprises operate the exact same way so it's important to select for a solution that can perform in your specific environment.

To address this concern, the following benchmark results will rely on standard datasets and compare the wide-table and multi-table query performance of StarRocks, Apache Druid®, ClickHouse, and Trino. Before examining the benchmark results, here's an overview of the wide-table and multi-table tests used in this report:

Wide-Table Testing with Star Schema Benchmark

Star Schema Benchmark (SSB) has been designed to test key performance metrics for various OLAP database products. SSB uses a star schema test set[1] that is widely applied in academia and industry.

Because ClickHouse flattens the star schema into a wide flat table and rewrites the SSB data into a flat table benchmark[2], we will be using the table creation method in ClickHouse to conduct our test on StarRocks, ClickHouse, and Apache Druid®.

Additionally, further performance testing will be conducted on the aggregation of low-cardinality fields.

Multi-Table Testing with TPC-H

TPC-H is a decision support benchmark developed by the Transaction Processing Performance Council (TPC). TPC-H can be used to build models based on real production environments to simulate the data warehouse of a sales system.

The main performance metrics are the response times of each query. The TPC-H benchmark evaluates a database system against the following capabilities:

- Examining large volumes of data.
- Execution of queries with a high degree of complexity.
- Providing answers to critical business questions[3].

Because ClickHouse and Apache Druid® are unable to finish the TPC-H test set, the test for multi-table scenarios considers only StarRocks and Trino.

Benchmark Results

Benchmark testing validates many of the claims made by StarRocks, Apache Druid®, ClickHouse, and Trino. These solutions are all incredibly fast in high-volume data scenarios. However, even with such high performance across the board, one solution performed particularly well.

Wide-Table Scenarios: StarRocks Offers 1.7x Performance Over ClickHouse and 2.2x Over Apache Druid®

Across the thirteen queries performed on the SSB flat tables, StarRocks achieved a response time 1.7x better than ClickHouse, and 2.2x faster than Apache Druid®.

With bitmap indexing and cache features enabled, StarRocks performed even better. This was especially true for queries Q2.2, Q2.3, and Q3.3. Overall, StarRocks delivered performance that was 2.2x that of ClickHouse and 2.9x that of Apache Druid® for this test.

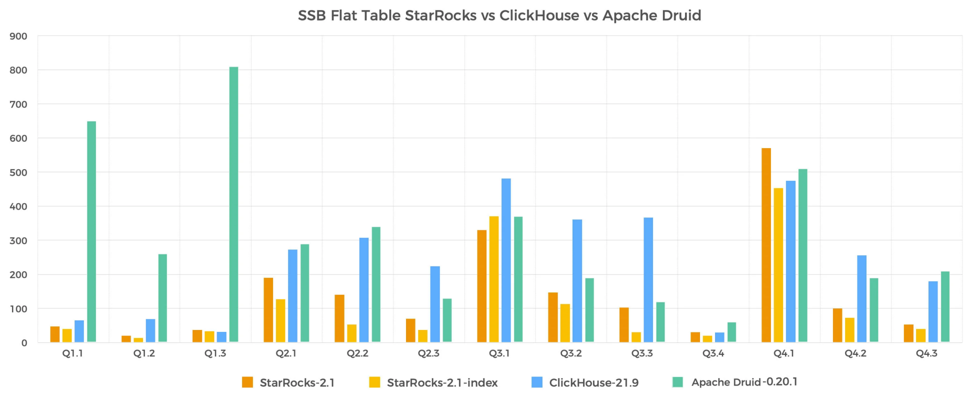


Figure 1: Wide-Table Performance

Query	StarRocks-2.1(ms)	StarRocks-2.1-index(ms)	ClickHouse-21.9(ms)	StarRocks Improvement Over ClickHouse	Druid-0.20.1(ms)	StarRocks Improvement Over Druid
Q1.1	47	40	65	1.38	650	13.83
Q1.2	20	13	69	3.45	260	13.00
Q1.3	37	33	31	0.84	810	21.89
Q2.1	190	127	273	1.44	290	1.53
Q2.2	140	53	307	2.19	340	2.43
Q2.3	70	37	224	3.20	130	1.86
Q3.1	330	370	481	1.46	370	1.12
Q3.2	147	113	361	2.46	190	1.29
Q3.3	103	30	367	3.56	120	1.17
Q3.4	30	20	29	0.97	60	2.00
Q4.1	570	453	475	0.83	510	0.89
Q4.2	100	73	256	2.56	190	1.90
Q4.3	53	40	180	3.40	210	3.96
SUM	**1837**	**1402**	**3118**	**1.70**	**4130**	**2.25**

Table 2: Wide-Table Query Results

Low-Cardinality Scenarios: StarRocks is 2.3x Faster Than ClickHouse and 3.2x Faster Than Apache Druid®

Because business use cases can vary, and because no single test can accurately reflect all common business use cases, several additional tests were run on scenarios where low-cardinality aggregation is frequently used (such as aggregation based on geographic location, age group, and product category).

Yet even under these circumstances, StarRocks performed exceptionally. Results showed a 2.3x query performance advantage for StarRocks over ClickHouse and a 3.2x advantage over Apache Druid®.

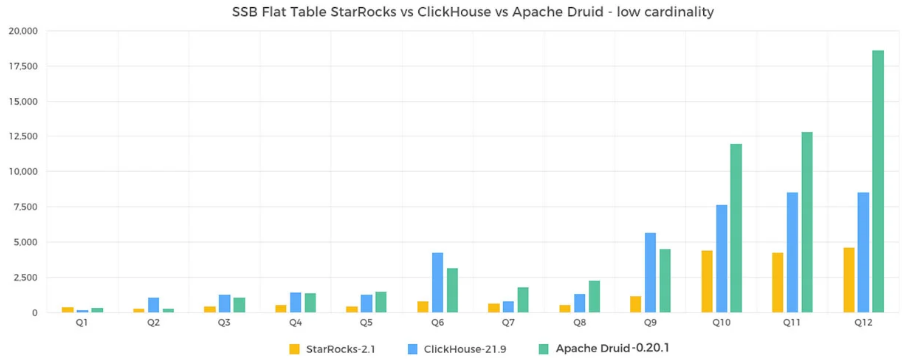


Figure 2: Low-Cardinality Performance

[TABLES AND CHARTS CONTINUE ON NEXT PAGE]

Query	SQL	Query type
Q1	select count(*),lo_shipmode from lineorder_flat group by lo_shipmode;	group by 1 low-cardinality column (< 50)
Q2	select count(distinct lo_shipmode) from lineorder_flat;	count distinct 1 low-cardinality column (< 50)
Q3	select count(*),lo_shipmode,lo_orderpriority from lineorder_flat group by lo_shipmode,lo_orderpriority;	group by 2 low-cardinality columns
Q4	select count(*),lo_shipmode,lo_orderpriority from lineorder_flat group by lo_shipmode,lo_orderpriority,lo_shippriority;	group by 2 low-cardinality columns and 1 int column
Q5	select count(*),lo_shipmode,s_city from lineorder_flat group by lo_shipmode,s_city;	group by 2 low-cardinality columns (7*250)
Q6	select count(*) from lineorder_flat group by c_city,s_city;	group by 2 low-cardinality columns (250*250)
Q7	select count(*) from lineorder_flat group by lo_shipmode,lo_orderdate;	group by 1 low-cardinality column (< 50) and 1 date column
Q8	select count(*) from lineorder_flat group by lo_orderdate,s_nation,s_region;	group by 2 low-cardinality columns (< 50) and 1 date column
Q9	select count(*) from lineorder_flat group by c_city,s_city,c_nation,s_nation;	group by 4 low-cardinality columns
Q10	select count(*) from (select count(*) from lineorder_flat group by lo_shipmode,lo_orderpriority,p_category,s_nation,c_nation) t;	group by 5 low-cardinality columns (< 50)
Q11	select count(*) from (select count(*) from lineorder_flat_distributed group by lo_shipmode,lo_orderpriority,p_category,s_nation,c_nation,p_mfgr) t;	group by 6 low-cardinality columns (< 50)
Q12	select count(*) from (select count(*) from lineorder_flat group by substr(lo_shipmode,2),lower(lo_orderpriority),p_category,s_nation,c_nation,s_region,p_mfgr) t;	group by 7 low-cardinality columns with function computation (<50)

Table 2: Low-Cardinality Query Details

Query	Result set cardinality	StarRocks (ms)	ClickHouse (ms)	StarRocks Improvement Over ClickHouse	Druid (ms)	StarRocks Improvement Over Druid
Q1	7	380	198	0.5	341	0.9
Q2	1	280	1055	3.8	304	1.1
Q3	35	470	1275	2.7	1072	2.3
Q4	35	550	1431	2.6	1391	2.5
Q5	1750	430	1273	3.0	1513	3.5
Q6	62500	790	4236	5.4	3146	4.0
Q7	16842	650	804	1.2	1825	2.8
Q8	60150	550	1357	2.5	2255	4.1
Q9	62500	1160	5683	4.9	4502	3.9
Q10	546875	4430	7635	1.7	11956	2.7
Q11	546875	4250	8540	2.0	12817	3.0
Q12	468750	4620	8538	1.8	18582	4.0
SUM		**18560**	**42025**	**2.3**	**59704**	**3.2**

Table 3: Low-Cardinality Query Results

Multi-table Scenarios: StarRocks Delivers 3.3x Better Query Performance Than Trino

Lastly, we performed a test on twenty-two queries against a TPC-H 100 GB dataset for both StarRocks and Trino. For these queries, StarRocks' native storage and its external tables for Apache Hive™ were used. StarRocks' Hive external tables and Trino were also used to query the same copy of data that was stored and compressed in the ORC and ZLIB formats respectively.

The results of this test show that StarRocks' overall query response time with native storage was 14.6x faster than that of Trino and 3.3x faster with StarRocks using external tables for Apache Hive.

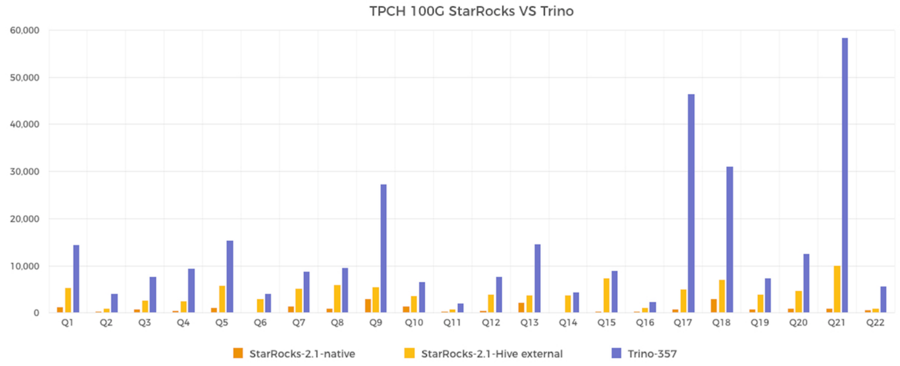


Figure 3: Multi-Table Performance

[TABLES AND CHARTS CONTINUE ON NEXT PAGE]

Query	StarRocks-native (ms)	StarRocks-Hive external (ms)	Trino (ms)	StarRocks-native Over Trino	StarRocks-Hive Over Trino
Q1	1291	5349	14350	11.1	2.7
Q2	257	940	4075	15.9	4.3
Q3	690	2585	7615	11.0	2.9
Q4	529	2514	9385	17.7	3.7
Q5	1142	5844	15350	13.4	2.6
Q6	115	2936	4045	35.2	1.4
Q7	1435	5128	8760	6.1	1.7
Q8	855	5989	9535	11.2	1.6
Q9	3028	5507	27310	9.0	5.0
Q10	1381	3654	6560	4.8	1.8
Q11	265	691	2090	7.9	3.0
Q12	387	3827	7635	19.7	2.0
Q13	2165	3733	14530	6.7	3.9
Q14	184	3824	4355	23.7	1.1
Q15	354	7321	8975	25.4	1.2
Q16	240	1029	2265	9.4	2.2
Q17	753	5011	46405	61.6	9.3
Q18	2909	7032	30975	10.6	4.4
Q19	702	3905	7315	10.4	1.9
Q20	963	4623	12500	13.0	2.7
Q21	870	10016	58340	67.1	5.8
Q22	545	991	5605	10.3	5.7
SUM	**21060**	**92449**	**307975**	**14.6**	**3.3**

Table 3: Multi-Table Query Results

Test configurations:

Hardware

- Host: 3 cloud hosts with the same configuration
- CPU: 16 cores
- Memory: 64 GB
- Network bandwidth: 5 Gbit/s
- Disk: ESSD

Software

- Kernel version: Linux 3.10.0-1127.13.1.el x86_64
- OS version: CentOS 7.8.2003
- Software version: StarRocks-2.1, Apache Druid® 0.20.1, ClickHouse 21.9, Trino-357, Apache Hive-3.1.2

StarRocks comes away as a clear performance winner in both our wide-table and multi-table testing, but this begs the question: what is driving the performance differences here in StarRocks' favor?

Let's take a deeper look at StarRocks and the capabilities that contributed to our benchmark results.

StarRocks Performance Advantages

Designed for speed and easy management, StarRocks incorporates a vectorized execution engine as well as an innovative Cost-Based Optimizer (CBO). Together, these features form the foundation for how StarRocks enables real-time data analytics and efficient queries on real-time mutable data.

StarRocks is also exceptionally flexible and can be used to build various models including wide tables, star schemas, and snowflake schemas. StarRocks further accelerates queries by allowing access to data that is stored in Hive, MySQL, and ElasticSearch through the use of external tables. Let's take a deeper look at each of these key features to understand exactly how they contribute to StarRocks' performance.

Vectorized Execution Engine

Compared to other solutions, StarRocks' vectorized execution engine makes more efficient use of CPUs thanks to how the engine stores and organizes data in memory and then computes SQL operators in a columnar manner. Columnar organization and computing make full use of the CPU cache and reduce the number of virtual function calls and branch predictions. All of this results in more efficient CPU instruction flows.

StarRocks' vectorized execution engine also makes full use of SIMD instructions and can complete more data operations with fewer instructions. Tests against standard datasets show that this engine enhances the overall performance of operators by 3 to 10 times.

In addition to vectorization, StarRocks has implemented other optimizations for the query engine. For example, StarRocks uses Operation on Encoded Data technology to directly execute operators on encoded strings without the need for decoding. This noticeably reduces SQL complexity and increases the query speed by more than double.

The engine can also be used for querying data from data lakes directly. StarRocks can directly query data stored in data lakes such as Hive, Iceberg, and Hudi. These data lakes are usually built on object storage (e.g. S3) or HDFS, and as can be seen in the benchmark results against Trino above, users can get an exceptional query experience without loading data into a data warehouse.

Cost-Based Optimizer

The performance of multi-table join queries is notoriously difficult to optimize. Execution engines alone cannot deliver ideal performance because the complexity of execution plans grow exponentially with the number of tables involved in the queries.

To address the above, a query optimizer that's intelligent enough to quickly identify the best query plan possible is required.

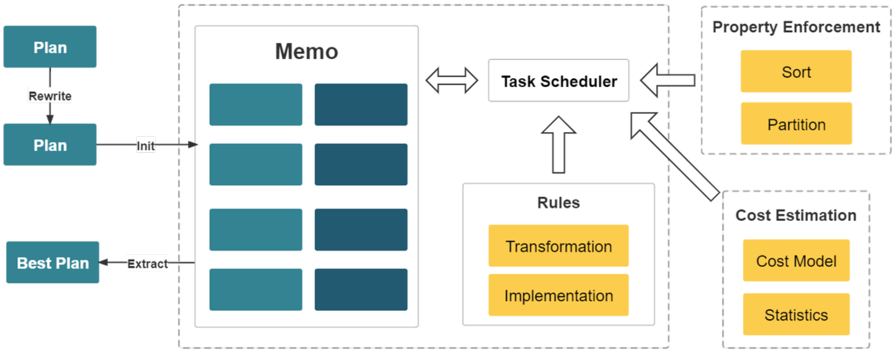


Figure 4: Cost-Based Optimizer Overview

In response to this, StarRocks has designed an innovative Cost-Based Optimizer (CBO). This CBO is Cascades-like and is deeply customized for StarRocks' vectorized execution engine with several optimizations and features. These optimizations include reusing common table expressions (CTEs), rewriting subqueries, Lateral Join, Join Reorder, strategy selection for distributed join execution, and streamlining for low cardinality. In total, the CBO supports all 99 TPC-DS SQL statements.

Moreover, StarRocks is continuously optimizing the HashJoin operator to leverage this brand-new CBO. The result is better multi-table join performance than other solutions, especially for complex multi-table join queries.

Materialized Views

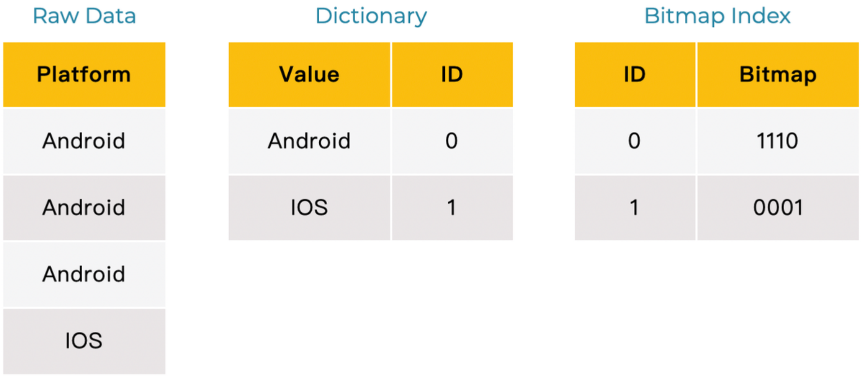
StarRocks also enables users to build intelligent materialized views across multiple tables. This helps users precompute metrics in advance which accelerates query performance and allows for higher concurrency. StarRocks' materialized views can also be automatically updated with changes made to the base tables, and when StarRocks conducts query planning, it will rewrite queries to fetch results from the appropriate materialized views.

These materialized views for multi-tables can greatly reduce the complexity of users' data pipelines and limit the number of ETL jobs. With StarRocks, users can leverage materialized views to build the result datasets they need. This is especially useful in cases where a simplified data pipeline is critical (like with real-time data analytics).

Additional Features for Query Acceleration

The features listed above contributed the lion's share of performance enhancements that drove StarRocks' results against ClickHouse, Apache Druid®, and Trino, but there are a few additional capabilities worth mentioning that StarRocks' performance benefits from:

Bitmap Index: StarRocks not only uses sorting to create primary indexes along with MinMax and Zone Maps for fast filtering, but it also supports bitmap-based secondary indexing. This offers great filtering capabilities for equal, non-equal, and range queries on the value column(s).



Raw Data

Platform
Android
Android
Android
IOS

Dictionary

Value	ID
Android	0
IOS	1

Bitmap Index

ID	Bitmap
0	1110
1	0001

Figure 5: Bitmap Example

Global Dictionary: StarRocks constructs a global dictionary for low-cardinality fields to convert string fields into integer fields which greatly improves the performance of count distinct and group by operations. To construct the global dictionary, users don't need to reload data or specify the dictionary while creating tables. StarRocks automatically builds global dictionaries for low-cardinality fields once the previous query is completed with query acceleration being transparent to users.

Lastly, StarRocks has a global runtime filter which effectively decreases the amount of data transferred between nodes during JOIN operations.

Summary

While it's clear solutions like Apache Druid®, ClickHouse, and Trino have their place serving the intense analytics needs of today's enterprises, for businesses seeking the best analytics performance on high data volumes, StarRocks stands out from the pack for both its speed and low maintenance costs.

The demand for real-time analytics shows no signs of slowing down, and as analytics becomes more democratized throughout the enterprise, this will place a tremendous burden on data engineering teams to figure out how to scale and accelerate their organization's analytics platform. For those organizations, solutions like StarRocks will make this stage of digital transformation a breeze.

References:

[1] Pat O'Neil, Betty O'Neil, Xuedong Chen: Star Schema Benchmark - https://www.cs.umb.edu/~poneil/StarSchemaB.PDF

[2] Star Schema Benchmark in ClickHouse documentation - https://clickhouse.com/docs/en/getting-started/example-datasets/star-schema/

[3] TPC Benchmark H Standard Specification Revision 3.0.0 - http://tpc.org/tpc_documents_current_versions/pdf/tpc-h_v3.0.0.pdf

About CelerData

CelerData enables enterprises to quickly and easily grow their business with a real-time analytical engine that is 3X the performance/cost of any other solutions on the market. CelerData is the only platform uniquely designed for the next generation real-time Enterprise, unleashing the power of business intelligence to help accelerate Enterprise digital transformation. Used worldwide by market leading brands including Airbnb, Lenovo and Trip.com, CelerData generates critical new insights for these data-driven companies. To learn more, please visit, www.celerdata.com