柏睿数据
BorayData

Technical White Paper

RapidsDB

Distributed In-memory Analytical Database

Version 4.X

Weixun BorayData Technology (Beijing) Co., Ltd.

Intelligent Data, Enabling Future!

No.19 Building, Poly International Plaza, No.7 Area of Wangjing East Garden,
Chaoyang District, Beijing
Telephone：+01064700868
Company Website：http://www.boraydata.cn

RapidsDB Rapids StreamDB Rapids ParallelAI AI work flow AI Rapids APU

Copyright

Disclaimer

Communication mode is as follows:

Weixun BorayData Technology (Beijing) Co., Ltd.
Room 503, No.19 Building (T1), Poly International Plaza, No.7 Area of Wangjing East Garden, Chaoyang District, Beijing
Telephone: +01064700868
Website: http://www.boraydata.cn

Trademark Declaration

is registered by Weixun BorayData Technology (Beijing) Co., Ltd. at Trademark Office of China National Intellectual Property Administration. Its exclusive use right is legally possessed by BorayData and protected by laws. Without written approval of BorayData, any part of the trademark shall not be used, copied, modified, transmitted, transcribed or bundled with other parts for sales by any organization or individual in any way. Any infringement of BorayData trademark right will be investigated legally by BorayData.

Version Declaration

Products involved herein are BorayData distributed in-memory analytical database (RapidsDB version 4.2.3) and the version of the relevant dependent packages is specified herein.

Version	Issue Date	Effective Date	Author	Reviewer	Approver	Current Status
V1.0	2020-11-03	2020-11-03	Arman	Mengju Zu	Mingxun Mo	Official
V1.1	2021-04-02	2021-04-02	Arman	Mengju Zu	Mingxun Mo	Official
V1.2	2021-04-16	2021-04-16	Arman	Mengju Zu	Mingxun Mo	Official
V1.3	2021-06-09	2021-06-09	Arman	Mengju Zu	Mingxun Mo	Official
						Official
						Official
						Official

Notes:

1. *When using, please update "cover" and "revision history" to the cover and revision history of this document, and delete the explanatory text.*
2. *V1.0 corresponds to 001 on the page footer, and V2.0 corresponds to 002, and so on.*
3. *If there is no reviewer and approver, please fill in the corresponding cell with "/".*

Contents

1 Scope

This document systematically introduces the core technology and the functional features of RapidsDB 4.0, a world-leading distributed in-memory database independently developed by BorayData. It explores many technical details of the product such as the high-performance and high-availability of the database, the massive data support, the application development environment, the operating environment and the management tools. These product features, functions and technical indicators intend to provide guidelines to database design, development and management personnel.

2 Overview

RapidsDB is a distributed in-memory relational database independently developed by BorayData Technology Co., Ltd., who owns the intellectual property of the product. It provides highly efficient and complete database management capabilities and fully supports online transaction processing (OLTP) and data warehouse (OLAP) to process transactions and analyze real-time data at scale. The database supports ANSI-standard SQL syntax and functions such as Aggregate, Group and Window to access, join, filter and analyze data.

RapdisDB provides a unified SQL access to various data sources, which include relational and non-relational data sources. Data can be integrated across all data sources. RapidsDB supports access to its internal in-memory data store, Hadoop HDFS CSV files and streaming data sources. It also provides JDBC interfaces to client applications.

RapidsDB 4.0 has the following key technical features:

- **High Performance**

  An enterprise's business growth can add a lot of pressure to concurrent data processing of a database. The distributed architecture of RapidsDB can leverage data clusters and partitions to process heavy workloads, maximizing the database performance. The current version of the database provides a variety of performance optimization methods, which include table structure optimization, data encoding, code generation, dynamic query, indexing, leaf nodes dynamic expansion, etc., to help enterprises handle transactions with heavy workloads.

- **High Reliability and High Availability**

  RapidsDB 4.0 provides a variety of technologies such as data backup and recovery and pairing mode to support the high reliability and availability of the database, which is required by the key business operation of an enterprise. These technologies guarantee that the database can provide 24/7 uninterrupted service.

- **RapidsDB Federation**

  The RapidsDB Federated Connector System is composed of a set of dynamic and pluggable connectors that can access the underlying data sources of the federated database. The Connectors manage the metadata of the objects (usually tables or files) that reside in the remote data stores and provide the metadata as an ANSI SQL-based schema to the RapidsDB Query Execution Engine. As the data is presented as a single federated database, the user can easily identify and combine the data using ANSI-standard SQL across any of the data sources. Through the RapidsDB Federation, RapidsDB can break data silos and easily query heterogenous data.

- **High Security**

RapidsDB requires authentications of all client connections before any commands can be issued. Similar to the federation of disparate data sources, RapidsDB can federate multiple authentication sources. User configuration can be used as a source for the authentication process. The current version supports password-based (internal authentication) and Kerberos-based user authentications.

- **Massive Data Processing**

  The current version of RapidsDB provides functions such as cluster dynamic expansion, data partitioning, data compression, columnar storage, etc. to support the massive data volume. Through the deployment of leaf nodes and data clusters, RapidsDB can process TB-level data sets while keeping CPU workloads and consumption under control.

- **High Manageability**

  The current version of the database integrates a newly designed web-based management platform, which can effectively reduce database management and maintenance costs and simplify cluster deployment and management.

3 Technical Architecture

The following figure shows the technical architecture of RapidsDB 4.0:

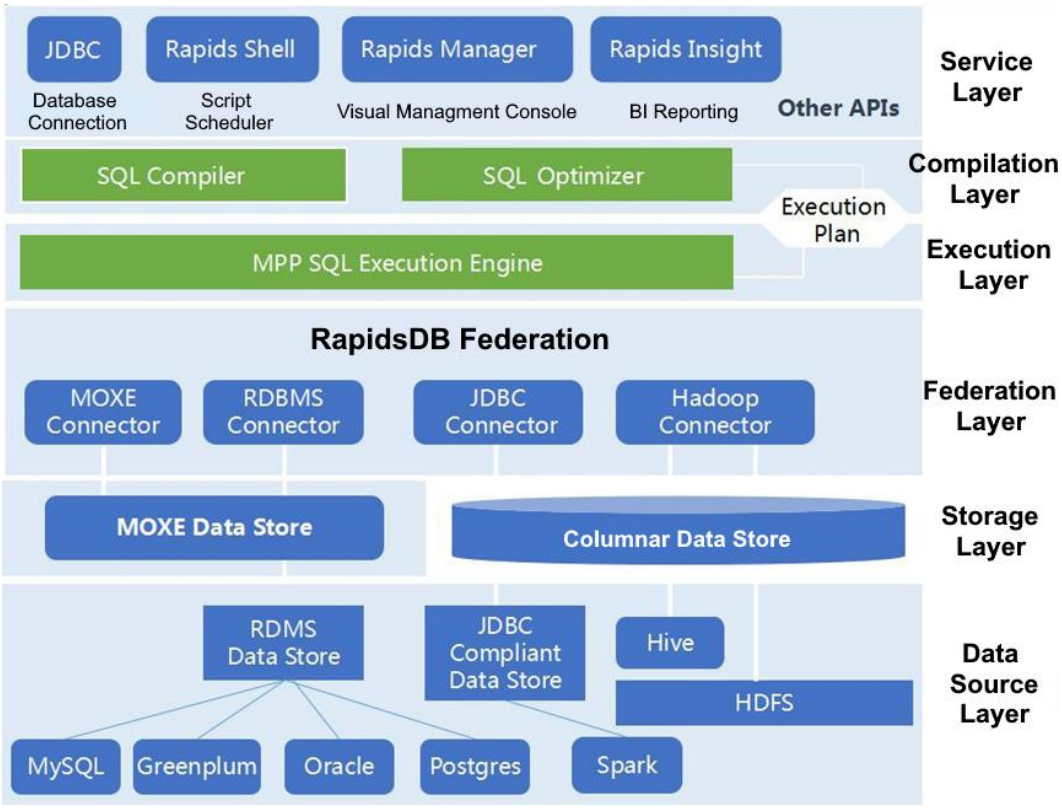

Figure 1 RapidsDB 4.0 Technical Architecture

From bottom to top, the technical architecture of RapidsDB consists of the following six layers:

- Data source layer: with the heterogeneous data query capability through Connectors, RapidsDB can access data that resides in external data sources, such as HDFS, Hive, MySQL, etc.
- Data storage layer: this layer includes two built-in storage engines, which are the distributed in-memory storage engine called MOXE and the memory-based columnar data storage engine, to realize functions of data storage, transactions, cache and log space management. Other functions, such as data partitioning, data compression and columnar storage, are also provided to optimize the database performance.
- Data federation layer: this layer creates Connectors and uniformly manages Connector functions
- Execution layer: RapidsDB has its own fully parallel MPP (Massively Parallel Processing) execution engine, which is responsible for executing query plans generated by the RapidsDB SQL Compiler and Optimizer. The MPP execution engine will access the underlying data sources through the RapidsDB Federation Connectors.
- Compilation layer: RapidsDB has an advanced SQL Compiler and Optimizer, which is responsible for query plan creation and SQL query

execution. Fully leveraging the native SQL functions of the underly data sources, the generated query plan analyzes and decides which operations can be pushed down to the data source to be executed directly. The MPP execution engine will execute the remaining parts of the query, only retrieving data from the required underlying data sources.

- Service layer: RapidsDB provides a command-line interface, Rapids-Shell, to configure Connectors and submit queries. It also integrates a web-based management console, RapidsDB Manager, for RapidsDB cluster configuration and management.

4 High Reliability

In the information technology industry, high reliability refers to an attribute of a system or a component that consistently performs according to its specifications, meeting the uptime expectation. Reliability can be expressed by the term of “100% operability” or “never failed”. A widely used but difficult- to-meet standard is the famous “Five Nines” standard, that is, the operational reliability should reach 99.999% of the time.

However, a computer system will inevitably encounter hardware failures, software malfunctions, operator errors or malicious attacks during operation. All of these problems may abnormally interrupt the ongoing transactions and cause data loss in some serious cases. Therefore, a database management system should have a reliable protection mechanism to enable a database to restore to the correct state.

RapidsDB provides complete backup and recovery solutions. Although it uses main memory to store data, copies of data are maintained on a server disk.

When RapidsDB runs in a fully persistent mode, transactions are committed to transaction logs on disk and compressed into database snapshots. The

directory used to store the snapshot and log files can be modified through the configuration file. Once the database restarts, it will load the snapshots and logs and restore itself to the state before being shut down.

Both snapshots and logs are in the internal binary format. Because RapidsDB will re-index when it restores or copies data, these files will only store row data without indexes. Without the indexes being stored, the size of a snapshot file is usually smaller than that of the corresponding used memory. A verification process is established to check both snapshot and log files to ensure the integrity of data. When an inconsistency is detected, the way to recover the snapshots and logs will be different. The database will temporarily enter the offline state, in which any connection to the database is disabled. The damaged part of a log will be automatically truncated and repaired. The database will load as much data as possible from the log into the memory and then change the database state to be online. As RapidsDB cannot guarantee the consistency of the data at that time, any damaged data in the log file cannot be recovered. If the operation of a damaged log is the same as that of future logs, it will give the wrong results when the damaged log is replayed. If RapidsDB finds an inconsistency during the verification process when snapshots are replayed, the database will enter the unrecoverable state. In that case, RESTORE command must be used to restore the database from a backup.

For data stored in memory, RapidsDB will asynchronously load data from snapshots and logs into memory when it starts. This is an automatic process. The recovery status of the database can be viewed by running the SHOW DATABASES command. For data stored in the columnar store on disk, data loss will not occur when the database restarts. To be safe, a single table or the entire database can be manually backed up and restored in a specified directory. The manual backup operation is applicable to both row-based and column-based tables.

5 High Availability

The high Availability refers to improving the ability of a system or an application to operate continuously by minimizing downtime caused by daily maintenance (planned) and sudden crashes (unplanned). It is different from the fault-tolerance technology, which guarantees the uninterrupted operation. A system with high availability is the most effective way to prevent a computer system shutdown due to failures.

Many important transactional systems in the banking, insurance, telecommunications industries and the government sector require databases to provide 24/7 operation. Therefore, database systems should maintain the high availability even in the events of a system failure, an application malfunction, a network error, a human error, etc.

The high availability mode of RapidsDB can be enabled in the environment variable, creating a primary and a secondary availability group. Cluster nodes share data copies in a paired configuration mode. A RapidsDB cluster is composed of multiple leaf nodes, which are responsible for the distributed data storage and computation. As the following figure shows, there is a one-to-one relationship between a computational leaf node in a primary availability group and a leaf node in a secondary availability group. During regular operation, Availability Group 1 is responsible for the data storage and computational workloads. Availability Group 2 is in the standby mode. When any node fails, a leaf node can automatically failover as needed. The cluster will not go offline. When the data in the primary partition changes, the user can synchronize the data in the replica partition by balancing data partitions, so as to maintain the data consistency of the primary and the secondary partitions.

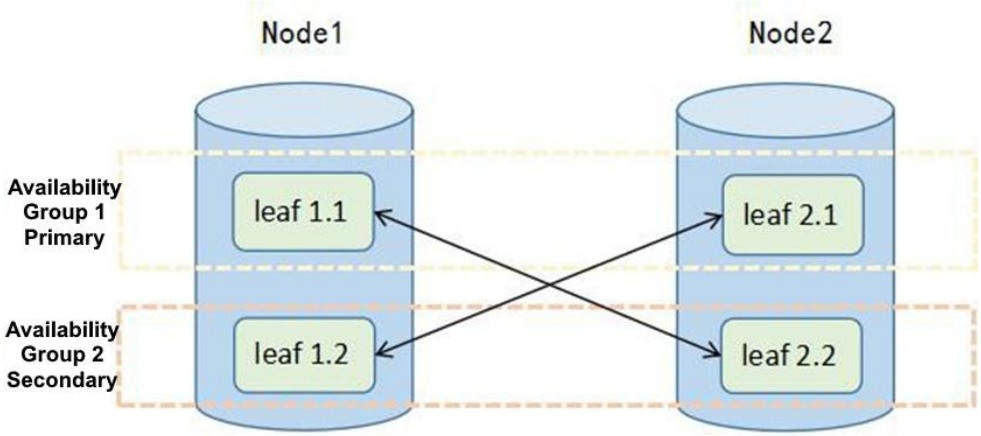

Figure 2 RapidsDB 4.0 Dual-Node High Availability

6 High Performance

With the exponential growth of business data, database performance issue has become more and more prominent. Database performance optimization has become a systematic project. The optimization scope should cover the tool layer, the interface layer, the transmission layer, the SQL layer, the storage layer and so forth. RapidsDB has provided the corresponding optimization measures as follows.

6.1 Data Partitioning

RapidsDB is a highly scalable distributed system. Data is automatically distributed among nodes in a cluster. Sharding can help to optimize the performance of distributed aggregate queries and filtered queries with equality predicates. Users can add nodes and re-shard the data as needed to expand the workloads.

When a user creates a database, the system will calculate a default data partitioning value based on the cluster size and the machine configuration. It represents the number of partitions the data will be divided into by default.

The user can manually define the number of partitions when the database is created. In the subsequent data insertion process, the database will partition each inserted data according to the partition key of the table structure defined by the user. Data with the same partition key value will be put into the same partition.

6.2 Data Storage

RapidsDB supports in-memory row-based data storage and memory-and-disk-based columnar data store. The in-memory row-based data storage provides the best real-time performance of transactional workloads. The memory-and-disk-based columnar data store is best for analytical workloads across large historical data sets. Columnar tables will be stored on disk, but transactions and caching will be done in memory. Most operations performed on disk are write-related, while most read-operations are completed in memory, which further improves the query efficiency.

The row-based storage is the most common data storage type of a relational database. Row-based data storage treats each row as a unit and stores the fields of a given row all at the same physical location. This makes row-based storage suitable for transactional workloads. With this data storage approach, selecting, inserting, updating or deleting a single row in a table usually will involve changes of most or all of the columns.

Columnar store treats each column as a unit and stores the data in each column at the same physical location. This approach creates two important advantages. One is that each column can be scanned separately. The database can essentially only scan the columns required by a query, keeping good caching locality during the scanning. The other advantage is that columnar storage is ideal for data compression. For instance, duplicated and similar data values can be easily compressed together.

The row-based storage table is the default table type when a table is created. The columnar storage table can be created by specifying an index type when a table is created. By combing the row-based data storage and the columnar data store approaches, a user can ingrate real-time and history data in a query, simplifying the technical stack.

6.3 Concurrent Queries

Concurrency control is the key technology of a database management system to guarantees database consistency, which is one of the important indicators to evaluate the performance of a database. RapidsDB's distributed Query Optimizer maximizes CPU efficiency by evenly distributing and processing a large number of workloads. Query plans are compiled into machine code and cached to accelerate subsequent queries. A key feature of these compiled query plans is that they don't have preset parameter values. RapidsDB can replace these values according to different requests so that subsequent queries with the same structure can be carried out quickly. In addition, by using MVCC and lock-free data structures, RapidsDB can maintain high data accessibility even in the case of a large number of concurrent reads and writes.

6.4 Data Encoding

RapidsDB supports several different types of data encoding to help compress data. All columnar store data is encoded in some way. Common data compression technologies include run-length encoding, dictionary encoding, LZ4, etc. These encoding operations, including filtering and aggregation, can run highly efficiently by enabling special encoding functions and the Single Instruction/Multiple Data (SIMD) instruction included in the Intel AVX2 processor's instruction set. By default, data encoding is

performed automatically. Users can benefit from the operation without changing any settings. The decision to enable data encoding is not made by the Query Optimizer but by the query execution system at run time.

6.5 Code Generation

RapidsDB's code generation system is one of the main contributing factors of the superior query execution performance. Compared with the traditional interpreter-based database execution model, RapidsDB embeds an industrial compiler to generate efficient machine code in order to achieve low-level optimization, which cannot be realized only by interpreting execution plans. By default, queries are interpreted first and then compiled asynchronously at the backend for future use. This approach speeds up complex query execution time by providing an efficient query plan for each future execution.

As mentioned above, when RapidsDB encounters a given query shape for the first time, it will asynchronously optimize and compile the query for future use. This approach brings the minimal overhead. It is not dependent on the amount of data that needs to be processed, but the complexity of the query. The code generation process will extract parameters from a query and then transform the normalized query into an intermediate representation specified by a single storage database and customized by the system. Subsequent requests with the same shape can reuse the plan to achieve rapid and consistent completion of the query.

Code generation is applicable to all data manipulation language (DML) queries. RapidsDB generates codes when running the CREATE TABLE and ALTER TABLE statements. Codes generated by these data definition language (DDL) queries can reduce the table compilation time of DML queries in the future.

6.6 Indexing

Same as all standard SQL databases, RapidsDB allows indexes to be created on tables, accelerating certain data access. The default indexing type of RapidsDB is skip list, which is intended to replace the B-tree index used by most databases including MySQL. A skip list is optimized to run in memory because it can be lock free and provide extremely fast insertion. Like B-tree, a skip list provides expected O(log(n)) lookup performance and can arrange sequential traversal.

Column indexing leverages the columnar storage technology to efficiently store and retrieve a large number of values from disk (flash memory or slid state disk is recommended). For other types of indexes, RapidsDB requires all data to be stored in memory. However, column indexing usually produces a large amount of compressed data supported by disk. Therefore, it is very useful for analytical workloads.

Generally, a hash index is used only if a query uses an equality filter for each indexed column. Therefore, hash indexes should be used only when measurable benefits of a user's specific data sets and workloads can be obtained. Under these special circumstances, hash indexes can be useful to match and access a unique value quickly and accurately.

7 Data Federation

RapidsDB Federated Connectors are a set of dynamic and pluggable connectors used to control the access to the underlying data stores that make up the federated database. The connectors manage the metadata of objects (usually tables or files) in remote data stores and provide the metadata as ANSI-based SQL schemas to the RapidsDB Query Execution Engine, enabling users to view the objects across multiple data sources as a single federated SQL database.

Connectors are responsible for the native data storage management and the data type conversion. The system will uniformly process data types in all data stores. Connectors provides a unified query interface independent of data sources, which enables ANSI-standard SQL queries and presents data to the RapidsDB Execution Engine in rows and columns.

In order to optimize performance, we generally want to have each underlying data source perform as much of the query as possible to reduce the amount of data that has to be pulled and processed by the RapidsDB Execution Engine. For example, when the data source is a relational database, it is typically capable of executing a join on the tables in that data source, or it can filter the database based on predicates in the query.

RapidsDB supports an optimization feature called “Adaptive Query Pushdown” to deal with this issue. With Adaptive Query Pushdown, each Connector involved in the execution of a federated query plan analyzes the plan and decides which operations can be pushed down to the data source and, for the remaining parts of the query, the optimum way to retrieve the data and transfer it to the RapidsDB Execution Engine to complete the query.

8 High Security

RapidsDB requires all client-end connections to prove user identities before issuing commands. Similar to how RapidsDB federates different data sources, it can federate multiple different authentication sources. Users can be configured and authenticated through a specific source, that is, an authenticator. The current version of RapidsDB supports user authentications through passwords (internal authentication) or the Kerberos (third-party authentication) authentication protocol.

8.1 Initial Users Setup

When RapidsDB starts for the first time, it will create an initial user. “RAPIDS” and “rapids” will be the user’s username and initial password, respectively. The initial user can use the information to authenticate his/her identity before configuring the system and creating other users. The RAPIDS user has more privileges than a regular user. The password of the initial user can be modified by executing the Modify command.

8.2 Authenticator

The authenticator is a pluggable module to control the authentication of a RapidsDB user. Each user of the system can only be associated with one authenticator instance. When a user attempts to be authenticated, the request will be processed and executed by the authenticator instance.

RapidsDB currently supports the internal password-based authentication and the third-party Kerberos authentication protocol. Except for the password-based authenticator, which is automatically created by the system, authenticator type instance needs to be created before the authenticator type can be used. Different authenticator type may require different information and configurations when it is created.

If an authenticator associated with a user has been deleted, the associated user can also be deleted afterwards as there is no authenticator, which can enable the user to log in again. Similarly, if an authenticator is disabled, all authentication requests it receives will be denied in order to block the logins of all the users associated with the authenticator instance. On the contrary, the internal password-based authenticator cannot be deleted or disabled.

9 Massive Data Processing

In recent years, global data is growing rapidly. This trend poses tremendous challenges for companies to manage a massive amount of data with traditional relational databases.

Leveraging the advanced technologies of data compression, data loading acceleration, data partitioning and so forth, RapidsDB has the capability to process massive data. It combines row-based and column-based data storage to support simultaneous access to both row store tables and columnar store tables with one single SQL query. When a user cannot load and store all data in memory due to limited budget, the columnar store provides a good hybrid data storage option. It can reduce the pressure on memory space, while optimizing a single node and improving query performance.

9.1 Data Compression

In a modern computer system, the I/O speed is far slower than the CPU processing speed, which results in the longest waiting time. Data compression technology can reduce storage cost. But more importantly, by taking a small amount of CPU, it can reduce a large amount of I/O frequency, greatly improving query processing speed. RapidsDB divides column data into different sorted groups by selected columnar store index keys. In each group, the duplicated column values will be compressed. In the following figure, the Price column is sorted and used as the index of the columnar store table.

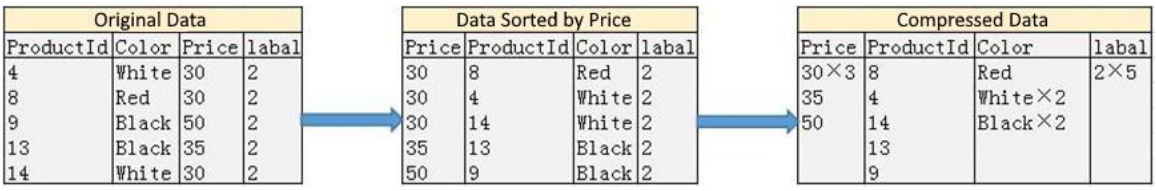

Figure 3 RapidsDB 4.0 Data Compression Workflow

9.2 Columnar Storage

Massive data processing mostly is associated with OLAP analytical scenarios. One of the characteristics of these scenarios is that they most often require batch processing. In that case, the unique features of data indexing, data partitioning and data compression of a columnar store can improve query efficiency to a certain extent. When querying a columnar store, only the associated columns are scanned, greatly reducing the amount of data that needs to be read. In addition, while handling massive data, some users cannot store all data in memory due to budget constrain. With the columnar storage approach, it offers the hybrid option of storing small tables in memory and big tables on disk, tremendously improving the massive data processing and analyzing capability.

10 Configuration Instructions

To install and use RapidsDB, hardware support is required. The following sections will list the RapidsDB configuration requirements as well as the recommendations for cluster deployment with the corresponding configurations for your reference.

10.1 Hardware Configuration

Items	Minimum Requirements	Recommended Requirements
CPU	2×2 Core	2×72 Core
Memory	4 GB	512 GB
Disk	100 GB	SSD 2 TB
Bandwidth	1000 MB	10 GB/25 GB
Operating System	CentOS/RH 6.5 or above	CentOS/RH 7.6 or above
Machine	Physical machine, virtual machine, Cloud platform	Physical machine, virtual machine, Cloud platform

10.1.1 CPU Configuration

A RapidsDB cluster consists of aggregator and leaf nodes. Each aggregator or leaf node is a process. Multiple aggregator and leaf nodes will consume certain

amount of CPU. As the CPU resource overhead can be very large when concurrent workloads are being processed, it is better to set up multi-core and multi-thread CPU configuration for a computing node when the hardware is configured.

10.1.2 Memory Configuration

RapidsDB is a distributed in-memory database. The memory capacity directly determines the amount of data that can be loaded into it. Therefore, the memory configuration should be based on the actual data volume of a project. If massive data needs to be processed, it is recommended to load hot data into memory and store cold data on disk. Considering that caching and temporary table storage will take up some memory space, it is better to reserve some space in advance when data is loaded.

10.1.3 Disk Configuration

When columnar storage is adopted, the disk capacity needs to be considered as well as certain disk I/O requirements. The data compression technology can save some storage space and reduce I/O frequency to improve the query performance. However, the regular data importing process requires disk I/O. As a result, a solid-state drive (SSD) is recommended. In addition, the disk will also be used to store data backups, logs and other files. Therefore, the disk size should be at least twice of the data size.

10.1.4 Bandwidth Configuration

As RapidsDB requires cluster deployment, communication between aggregator and leaf nodes is dependent on the bandwidth to ensure the data transmission efficiency. An optical fiber network environment is recommended to handle massive data transactions and big clusters so that the maximum performance of the database can be guaranteed.

10.1.5 Operating System Configuration

RapidsDB can be installed and deployed on many mainstream operating systems such as CentOS, Red Hat, etc. as well as a series of domestic

operating systems such as KylinOS, Hongqi Linux, Ningsi, etc.

10.2 Cluster Configuration

10.2.1 Cluster Size

A RapidsDB Cluster has a master aggregator (master node, which only exists on a master server ）to manage other sub aggregators（ slave node, each server can have many slave nodes）. Each sub aggregator on a server can have multiple leaf nodes.

The number of aggregators depends on each specific uses case. For example, if a cluster is used for multiple types of workloads (for instance, it is the backend of a web application as well as the object of an analytical query), it is recommended to have multiple aggregators or an aggregation pool for these independent workloads.

Besides workload distribution, the most important consideration should be network bandwidth. As a rule of thumb, a cluster with fifty or fewer nodes should have a leaf to aggregator node ratio of 5:1. A cluster with more than fifty nodes, the ratio should be 10:1. Please note that you can add nodes to a cluster after the cluster is up and running in order to adjust the overall performance.

The appropriate ratio of aggregator and leaf nodes also depends on the type of the operating workloads. Transactional workloads that run many small queries or queries involving only a single partition require more aggregator nodes because each query interacts with an aggregator and a leaf node, respectively. Analytical workloads, especially those involving distributed connections, require fewer aggregator nodes because almost all work is done on leaf nodes.

10.2.2 System Limitations

- Cluster Limitations

Description	Maximum Size	Notes
Number of Hosts	Unlimited	Only clusters with a limited number of hosts will have workloads that generate large amounts of network traffic among the hosts
Number of Nodes	Unlimited	Only clusters with a limited number of hosts will have workloads that generate large amounts of network traffic among the hosts
Number of Databases	Unlimited	Each database uses memory to establish a transaction buffer. More databases will demand more memory.
Number of Concurrent Queries	Unlimited	Each aggregator node only executes a limited number of queries (192 by default) at a time. Performance improves with the growth of the cluster size.
Number of Simultaneous Connections	100,000/aggregator	

- Database Limitations

Description	Maximum Size	Notes
Number of Tables	Unlimited	Each table consumes certain amount of memory overhead
Number of Partitions	Unlimited	Each user can define the number of partitions

- Table Limitations

Description	Maximum Size	Notes
Number of columns in a table	4096	
Row size in byes	64 KB	The restriction does not apply to variable length strings, such as VARCHAR, VARBINARY, TEXT,etc.
Character name length (applicable to table name and column name)	64	

Number of indexes in a table	63	
Number of columns for an index	32	
Number of joined tables in a Select statement	253	

11 High Manageability

Rapids Manager is a web-based database management console with a visual interface to provide powerful management capabilities, which include RapidsDB service management, cluster monitoring, operation and maintenance, etc. The easy-to-use console automates cluster optimization based on different hardware environments, providing the best big data analytics service to customers with the optimal performance and reliability. It has the following features:

- Bare metal, virtual installation and fast deployment
- Rapids Stack service management GUI
- Automatic optimization of RapidsDB Cluster
- Supports secondary development
- GUI for query development and execution
- Provides system-level monitoring of various RapidsDB components

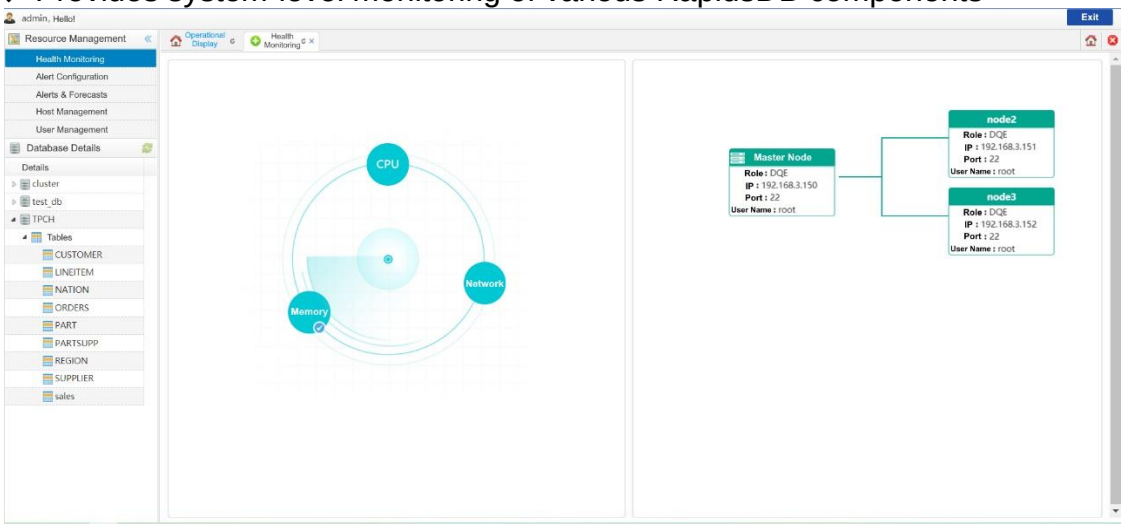

Figure 4 Rapids Manager- Cluster Monitoring

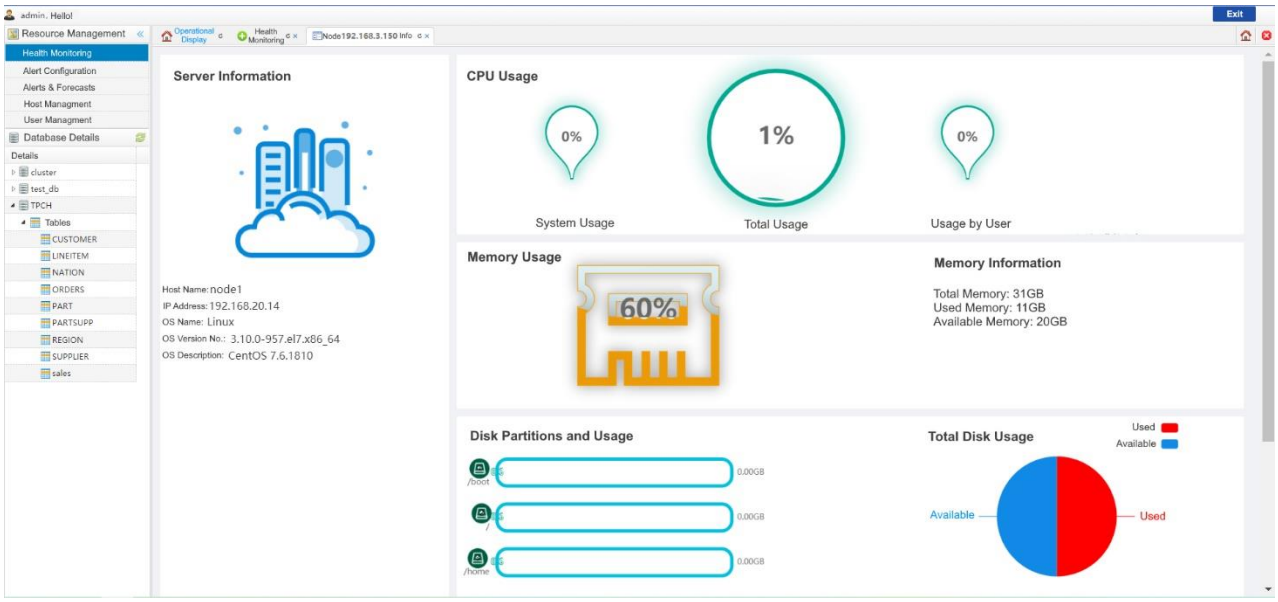

Figure5 Rapids Manager - Nodes Monitoring

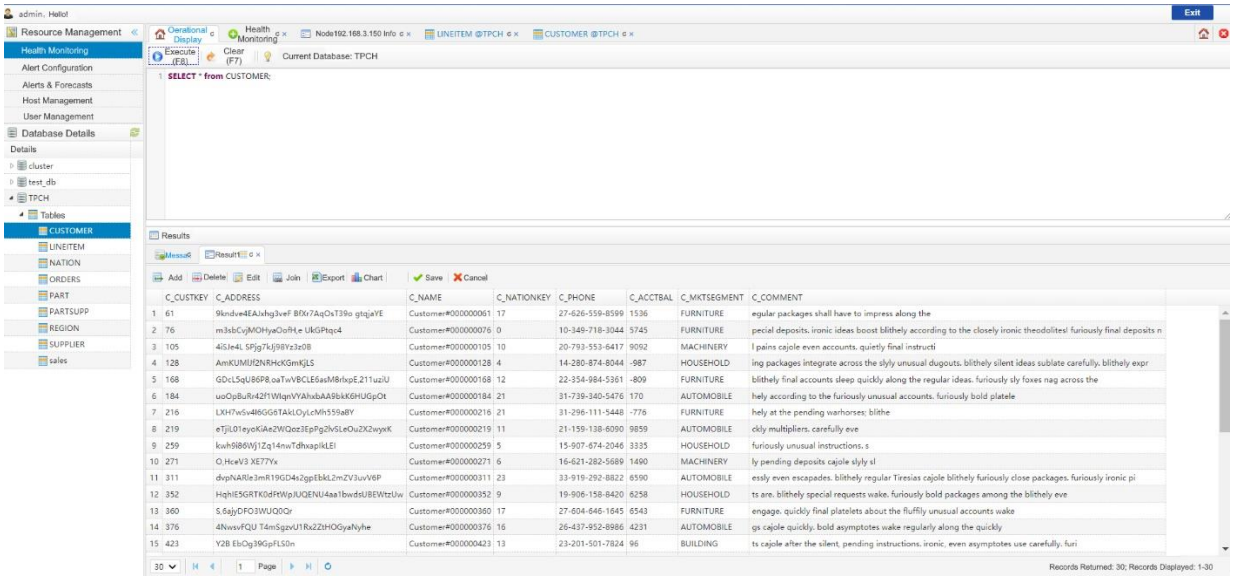

Figure6 Rapids Manager - Database User Interface

12 BorayData Technical Support

12.1 Standard Service

Within the guaranteed product service period：

1. Provide customers with online support via telephone, email,

WeChat, etc.

2. Provide customers with 5x9 online support to have technical experts respond to online tech support questions via telephone, email, WeChat, etc.
3. Provide product patching service
4. Provide patch fixes service when necessary
5. Provide software upgrade package
6. Provide the upgraded version of the software with new functions
7. Provide software installation and deployment services
8. Send tech engineers to install and deploy the software on site based on the actual needs of a project. Design installation and configuration plan and provide parameter optimization service based on the operating environment.
9. Provide product documentation to help customer understand the product functions and features
10. Provide product installation manual and testing guidelines
11. Revisit customers and provide inspection service according to the agreement of both parties (optional)
12. Provide customer training service (optional)
13. Provide BorayData product and technical training according to the agreement of both parties

12.2 Partner Services

BorayData appreciates the collaboration with partners and provides an excellent reward program which includes:

1. Systematic training support, including sales and technical training
2. Pre-sales solution support to important projects
3. POC and performance optimization support to important projects
4. Timely sharing of the latest successful domestic use cases
5. Potential project collaboration opportunities sharing
6. Certification training and issuance

13 Company Profile

Weixun BorayData Technology (Beijing) Co., Ltd. (BorayData) was founded in 2014. It is a world-leading real-time big data analytics provider of distributed in-memory database technology, international standard-based database products with intellectual property rights, and industrial database solutions. The company is committed to the research and development of distributed in-memory OLAP databases, providing real-time big data analytics products and services to support the digital transformation of various industries and government sectors.

The advanced database technology establishes the core competitiveness of BorayData. The company is headquartered in Beijing and has global R&D centers in Silicon Valley of USA, Sidney of Australia, Shanghai, Guangzhou, Wuhan and many cities in China. The R&D team is led by Liu Ruiming, founder and chief architect of the company, and some best international database experts. Among them, three core technical engineers with more than twenty years of database research and development experience are students of Jim Gray, a Turing Award winner and a database technology pioneer.

BorayData has made breakthroughs in database core technology innovation, product research and development, international database standard establishment and industrial application integration since the company was established five years go. Beginning writing the first line of codes, BoreyData has independently developed six database core products, which include the distributed in-memory database, the streaming database, the embedded artificial intelligence parallel algorithm library, the heterogenous data federation connectors, the in-memory data storage engine and the data transmission acceleration and security chip. From the SQL parsing layer, the optimization layer, the execution layer to the database storage engine, the Rapids Data Platform (RDP), a real-time big

data analytics platform, is fully controllable.

With the support of the Ministry of Industry and Information Technology,

BorayData actively participated in the creation of international database standards. Selected as the Chinese enterprise representative of the ISO members and supported by independently developed core database products, BorayData has created two international database standards: “SQL9075 2018 Streaming Database” and “AI-in-Database artificial intelligence”, making a groundbreaking achievement of China to voice its opinion and take the lead to establish international database standards.

BorayData focuses on big data use cases involving structural diversification, high throughput and high concurrency. From data collection, aggregation, mining to value realization, BorayData has built a highly available and flexible big data platform with dynamic scalability for business and government customers, meeting their demands for processing and analyzing massive heterogenous data in real time accurately and securely, deep learning, ad hoc queries, etc. It helps customers monetize the true value of data. Currently RapidsDB has been successfully implemented in various fields such as governmental macroeconomy, social and public opinion, financial services, telecommunications, industrial energy, etc. The product has received many awards, which include the “ Key Secure and Controllable Database Product for National Defense System”, the“ 2018 Big Data Industry Pilot Demonstration Project of the Ministry of Industry and Information Technology” and the “Best Use Case of 2018 Software Expo”.