TiDB Release Notes: Changes from v7.2.0 to v7.5.0

PingCAP

20231201

Abstract

This document contains the release notes for TiDB v7.2.0-DMR, v7.3.0-DMR, v7.4.0-DMR, and v7.5.0-LTS. When you upgrade from v7.1.x to v7.5.0, you can refer to this document for a thorough overview of new features, compatibility changes, improvements, and bug fixes.

For detailed guidance and additional resources about TiDB v7.5.0, see TiDB v7.5 documentation.

Table of Contents

1 TiDB 7.5.0 Release Notes

Release date: December 1, 2023

TiDB version: 7.5.0

Quick access: Quick start | Production deployment | Installation packages

TiDB 7.5.0 is a Long-Term Support Release (LTS).

Compared with the previous LTS 7.1.0, 7.5.0 includes new features, improvements, and bug fixes released in 7.2.0-DMR, 7.3.0-DMR, and 7.4.0-DMR. When you upgrade from 7.1.x to 7.5.0, you can download the TiDB Release Notes PDF to view all release notes between the two LTS versions. The following table lists some highlights from 7.2.0 to 7.5.0:

Category	Feature	Description
Scalability and Performance	Support running multiple ADD INDEX statements in parallel	This feature allows for concurrent jobs to add multiple indexes for a single table. Previously, it would take the time of X plus the time of Y to execute two ADD INDEX statements simultaneously (X and Y). With this feature, adding two indexes X and Y in one SQL can be concurrently executed, and the total execution time of DDL is significantly reduced. Especially in scenarios with wide tables, internal test data shows that performance can be improved by up to 94%.
Reliability and Availability	Optimize Global Sort (experimental, introduced in v7.4.0)	TiDB v7.2.0 introduced the distributed execution framework. For tasks that take advantage of this framework, v7.4 introduces global sorting to eliminate the unnecessary I/O, CPU, and memory spikes caused by temporarily out-of-order data during data re-organization tasks. The global sorting takes advantage of external

Category	Feature	Description
		shared object storage (Amazon S3 in this first iteration) to store intermediary files during the job, adding flexibility and cost savings. Operations like ADD INDEX and IMPORT INTO will be faster, more resilient, more stable, more flexible, and cost less to run.
	Resource control for background tasks (experimental, introduced in v7.4.0)	In v7.1.0, the Resource Control feature was introduced to mitigate resource and storage access interference between workloads. TiDB v7.4.0 applied this control to the priority of background tasks as well. In v7.4.0, Resource Control now identifies and manages the priority of background task execution, such as auto-analyze, Backup & Restore, bulk load with TiDB Lightning, and online DDL. In future releases, this control will eventually apply to all background tasks.
	Resource control for managing runaway queries (experimental, introduced in v7.2.0)	Resource Control is a framework for resource-isolating workloads by Resource Groups, but it makes no calls on how individual queries affect work inside of each group. TiDB v7.2.0 introduces "runaway queries control" to let you control how TiDB identifies and treats these queries per Resource Group. Depending on needs, long running queries might be terminated or throttled, and the queries can be identified by exact SQL text, SQL digests or their plan digests, for

Category	Feature	Description
		better generalization. In v7.3.0, TiDB enables you to proactively watch for known bad queries, similar to a SQL blocklist at the database level.
SQL	MySQL 8.0 compatibility (introduced in v7.4.0)	In MySQL 8.0, the default character set is utf8mb4, and the default collation of utf8mb4 is utf8mb4_0900_ai_ci. TiDB v7.4.0 adding support for this enhances compatibility with MySQL 8.0 so that migrations and replications from MySQL 8.0 databases with the default collation are now much smoother.
DB Operations and Observability	TiDB Lightning's physical import mode integrated into TiDB with IMPORT INTO (GA)	Before v7.2.0, to import data based on the file system, you needed to install TiDB Lightning and used its physical import mode. Now, the same capability is integrated into the IMPORT INTO statement so you can use this statement to quickly import data without installing any additional tool. This statement also supports the distributed execution framework for parallel import, which improves import efficiency during large-scale imports.
	Specify the respective TiDB nodes to execute the ADD INDEX and IMPORT INTO SQL statements (GA)	You have the flexibility to specify whether to execute ADD INDEX or IMPORT INTO SQL statements on some of the existing TiDB nodes or newly added TiDB nodes. This approach enables resource isolation from the rest of the TiDB nodes, preventing any impact on business operations while ensuring

Category	Feature	Description
		optimal performance for executing the preceding SQL statements. In v7.5.0, this feature becomes generally available (GA).
	DDL supports pause and resume operations (GA)	Adding indexes can be big resource consumers and can affect online traffic. Even when throttled in a Resource Group or isolated to labeled nodes, there may still be a need to suspend these jobs in emergencies. As of v7.2.0, TiDB now natively supports suspending any number of these background jobs at once, freeing up needed resources while avoiding having to cancel and restart the jobs.
	TiDB Dashboard supports heap profiling for TiKV	Previously, addressing TiKV OOM or high memory usage issues typically required manual execution of jeprof to generate a heap profile in the instance environment. Starting from v7.5.0, TiKV enables remote processing of heap profiles. You can now directly access the flame graph and call graph of heap profile. This feature provides the same simple and easy-to-use experience as Go heap profiling.

1.1 Feature details

1.1.1 Scalability

- Support designating and isolating TiDB nodes to distributedly execute ADD INDEX or IMPORT INTO tasks when the distributed execution framework is enabled #46258 @ywqzzy

  Executing ADD INDEX or IMPORT INTO tasks in parallel in a resource-intensive cluster can consume a large amount of TiDB node resources, which can lead to cluster performance degradation. To

avoid performance impact on existing services, v7.4.0 introduces the system variable tidb_service_scope as an experimental feature to control the service scope of each TiDB node under the TiDB backend task distributed execution framework. You can select several existing TiDB nodes or set the TiDB service scope for new TiDB nodes, and all distributedly executed ADD INDEX and IMPORT INTO tasks only run on these nodes. In v7.5.0, this feature becomes generally available (GA).

For more information, see documentation.

1.1.2 Performance

- The TiDB backend task distributed execution framework becomes generally available (GA), improving the performance and stability of ADD INDEX and IMPORT INTO tasks in parallel execution #45719 @wjhuang2016

  The backend task distributed execution framework introduced in v6.6.0 has become GA. In versions before TiDB v7.1.0, only one TiDB node can execute DDL tasks at the same time. Starting from v7.1.0, multiple TiDB nodes can execute the same DDL task in parallel under the backend task distributed execution framework. Starting from v7.2.0, the backend task distributed execution framework supports multiple TiDB nodes to execute the same IMPORT INTO task in parallel, thereby better utilizing the resources of the TiDB cluster and significantly improving the performance of DDL and IMPORT INTO tasks. In addition, you can also increase TiDB nodes to linearly improve the performance of these tasks.

  To use the backend task distributed execution framework, set tidb_enable_dist_task value to ON.

  ```
  SET GLOBAL tidb_enable_dist_task = ON;
  ```

  For more information, see documentation.

- Improve the performance of adding multiple indexes in a single SQL statement #41602 @tangenta

  Before v7.5.0, when you add multiple indexes (ADD INDEX) in a single SQL statement, the performance was similar to adding multiple indexes using separate SQL statements. Starting from v7.5.0, the performance of adding multiple indexes in a single SQL statement is

significantly improved. Especially in scenarios with wide tables, internal test data shows that performance can be improved by up to 94%.

1.1.3 DB operations

- DDL jobs support pause and resume operations (GA) #18015 @godouxm

  The pause and resume operations for DDL jobs introduced in v7.2.0 become generally available (GA). These operations let you pause resource-intensive DDL jobs (such as creating indexes) to save resources and minimize the impact on online traffic. When resources permit, you can seamlessly resume DDL jobs without canceling and restarting them. This feature improves resource utilization, enhances user experience, and simplifies the schema change process.

  You can pause and resume multiple DDL jobs using ADMIN PAUSE DDL JOBS or ADMIN RESUME DDL JOBS:

  ```
  ADMIN PAUSE DDL JOBS 1,2;
  ADMIN RESUME DDL JOBS 1,2;
  ```

  For more information, see documentation.

- BR supports backing up and restoring statistics #48008 @Leavrth

  Starting from TiDB v7.5.0, the br command-line tool introduces the --ignore-stats parameter to back up and restore database statistics. When you set this parameter to false, the br command-line tool supports backing up and restoring statistics of columns, indexes, and tables. In this case, you do not need to manually run the statistics collection task for the TiDB database restored from the backup, or wait for the completion of automatic collection tasks. This feature simplifies database maintenance work and improves query performance.

  For more information, see documentation.

1.1.4 Observability

- TiDB Dashboard supports heap profiling for TiKV #15927 @Connor1996

Previously, addressing TiKV OOM or high memory usage issues typically required manual execution of jeprof to generate a heap profile in the instance environment. Starting from v7.5.0, TiKV enables remote processing of heap profiles. You can now directly access the flame graph and call graph of heap profile. This feature provides the same simple and easy-to-use experience as Go heap profiling.

For more information, see documentation.

1.1.5 Data migration

- Support the IMPORT INTO SQL statement (GA) #46704 @D3Hunter

  In v7.5.0, the IMPORT INTO SQL statement becomes generally available (GA). This statement integrates the Physical Import Mode capability of TiDB Lightning and allows you to quickly import data in formats such as CSV, SQL, and PARQUET into an empty table in TiDB. This import method eliminates the need for a separate deployment and management of TiDB Lightning, thereby reducing the complexity of data import and greatly improving import efficiency.

  For more information, see documentation.

- Data Migration (DM) supports blocking incompatible (data-consistency-corrupting) DDL changes #9692 @GMHDBJD

  Before v7.5.0, the DM Binlog Filter feature can only migrate or filter specified events, and the granularity is relatively coarse. For example, it can only filter large granularity of DDL events such as ALTER. This method is limited in some scenarios. For example, the application allows ADD COLUMN but not DROP COLUMN, but they are both filtered by ALTER events in the earlier DM versions.

  To address such issues, v7.5.0 refines the granularity of the supported DDL events, such as support filtering MODIFY COLUMN (modify the column data type), DROP COLUMN, and other fine-grained DDL events that lead to data loss, truncation of data, and loss of precision. You can configure it as needed. This feature also supports blocking incompatible DDL changes and reporting errors for such changes, so that you can intervene manually in time to avoid impacting downstream application data.

  For more information, see documentation.

- Support real-time checkpoint updates for continuous data validation #8463 @lichunzhu

  Before v7.5.0, the continuous data validation feature ensures the data consistency during replication from DM to downstream. This serves as the basis for cutting over business traffic from the upstream database to TiDB. However, due to various factors such as replication delay and waiting for re-validation of inconsistent data, the continuous validation checkpoint must be refreshed every few minutes. This is unacceptable for some business scenarios where the cutover time is limited to tens of seconds.

  With the introduction of real-time updating of checkpoint for continuous data validation, you can now provide the binlog position from the upstream database. Once the continuous validation program detects this binlog position in memory, it immediately refreshes the checkpoint instead of refreshing it every few minutes. Therefore, you can quickly perform cut-off operations based on this immediately updated checkpoint.

  For more information, see documentation.

1.2 Compatibility changes

Note:

This section provides compatibility changes you need to know when you upgrade from v7.4.0 to the current version (v7.5.0). If you are upgrading from v7.3.0 or earlier versions to the current version, you might also need to check the compatibility changes introduced in intermediate versions.

1.2.1 System variables

Variable name	Change type	Description
tidb_enable_fast_analyze	Deprecated	Controls whether to enable the statistics Fast Analyze feature. This feature is deprecated in v7.5.0.
tidb_analyze_partition_concurrency	Modified	Changes the default value from 1 to 2 after further tests.
tidb_build_stats_concurrency	Modified	Changes the default value from 4 to 2 after further tests.

Variable name	Change type	Description
tidb_merge_partition_stats_concurrency	Modified	This system variable takes effect starting from v7.5.0. It specifies the concurrency of merging statistics for a partitioned table when TiDB analyzes the partitioned table.
tidb_build_sampling_stats_concurrency	Newly added	Controls the sampling concurrency of the ANALYZE process.
tidb_enable_async_merge_global_stats	Newly added	This variable is used by TiDB to merge statistics asynchronously to avoid OOM issues.
tidb_gogc_tuner_max_value	Newly added	Controls the maximum value of GOGC that the GOGC Tuner can adjust.
tidb_gogc_tuner_min_value	Newly added	Controls the minimum value of GOGC that the GOGC Tuner can adjust.

1.2.2 Configuration file parameters

Configuration file	Configuration parameter	Change type	Description
TiKV	raftstore.region-compact-min-redundant-rows	Modified	Sets the number of redundant MVCC rows required to trigger RocksDB compaction. Starting from v7.5.0, this configuration item takes effect for the "raft-kv" storage engine.
TiKV	raftstore.region-compact-redundant-rows-percent	Modified	Sets the percentage of redundant MVCC rows required to trigger RocksDB compaction. Starting from v7.5.0, this configuration item takes effect for the "raft-kv" storage engine.
TiKV	raftstore.evict-cache-on-memory-ratio	Newly added	When the memory usage of TiKV exceeds 90% of the system available memory, and the memory occupied by Raft entry cache exceeds the evict-cache-on-memory-ratio of used memory, TiKV evicts the Raft entry cache.
TiKV	memory.enable-heap-profiling	Newly added	Controls whether to enable Heap Profiling to track the memory usage of TiKV.

Configuration file	Configuration parameter	Change type	Description
TiKV	memory.profiling-sample-per-bytes	Newly added	Specifies the amount of data sampled by Heap Profiling each time, rounding up to the nearest power of 2.
BR	--ignore-stats	Newly added	Controls whether to back up and restore database statistics. When you set this parameter to false, the br command-line tool supports backing up and restoring statistics of columns, indexes, and tables.
TiCDC	case-sensitive	Modified	Changes the default value from true to false after further tests, which means that the table names and database names in the TiCDC configuration file are case-insensitive by default.
TiCDC	sink.dispatchers.partition	Modified	Controls how TiCDC dispatches incremental data to Kafka partitions. v7.5.0 introduces a new value option columns, which uses the explicitly specified column values to calculate the partition number.
TiCDC	sink.column-selectors	Newly added	Controls the specified columns of data change events that TiCDC sends to Kafka when dispatching incremental data.
TiCDC	sql-mode	Newly added	Specifies the SQL mode used by TiCDC when parsing DDL statements. The default value is the same as the default SQL mode of TiDB.
TiDB Lightning	--importer	Deleted	Specifies the address of TiKV-importer, which is deprecated in v7.5.0.

1.3 Offline package changes

Starting from v7.5.0, the following contents are removed from the TiDB-community-toolkit binary package:

- tikv-importer-{version}-linux-{arch}.tar.gz
- mydumper
- spark-{version}-any-any.tar.gz
- tispark-{version}-any-any.tar.gz

1.4 Deprecated features

- Mydumper is deprecated in v7.5.0 and most of its features have been replaced by Dumpling. It is strongly recommended that you use Dumpling instead of Mydumper.
- TiKV-importer is deprecated in v7.5.0. It is strongly recommended that you use the Physical Import Mode of TiDB Lightning as an alternative.
- Starting from TiDB v7.5.0, technical support for the data replication feature of TiDB Binlog is no longer provided. It is strongly recommended to use TiCDC as an alternative solution for data replication. Although TiDB Binlog v7.5.0 still supports the Point-in-Time Recovery (PITR) scenario, this component will be completely deprecated in future versions. It is recommended to use PITR as an alternative solution for data recovery.
- The Fast Analyze feature (experimental) for statistics is deprecated in v7.5.0.
- The incremental collection feature (experimental) for statistics is deprecated in v7.5.0.

1.5 Improvements

- TiDB
  - Optimize the concurrency model of merging GlobalStats: introduce tidb_enable_async_merge_global_stats to enable simultaneous loading and merging of statistics, which speeds up the generation of GlobalStats on partitioned tables. Optimize the memory usage of merging GlobalStats to avoid OOM and reduce memory allocations. #47219 @hawkingrei
  - Optimize the ANALYZE process: introduce tidb_build_sampling_stats_concurrency to better control the ANALYZE concurrency to reduce resource consumption. Optimize the memory usage of ANALYZE to reduce memory allocation and avoid frequent GC by reusing some intermediate results. #47275 @hawkingrei
  - Optimize the use of placement policies: support configuring the range of a policy to global and improve the syntax support for common scenarios. #45384 @nolouch

  - Improve the performance of adding indexes with tidb_ddl_enable_fast_reorg enabled. In internal tests, v7.5.0 improves the performance by up to 62.5% compared with v6.5.0. #47757 @tangenta

- TiKV
  - Avoid holding mutex when writing Titan manifest files to prevent affecting other threads #15351 @Connor1996
- PD
  - Improve the stability and usability of the evict-slow-trend scheduler #7156 @LykxSassinato
- Tools
  - Backup & Restore (BR)
    - Add a new inter-table backup parameter table-concurrency for snapshot backups. This parameter is used to control the inter-table concurrency of meta information such as statistics backup and data validation #48571 @3pointer
    - During restoring a snapshot backup, BR retries when it encounters certain network errors #48528 @Leavrth

1.6 Bug fixes

- TiDB
  - Prohibit split table operations on non-integer clustered indexes #47350 @tangenta
  - Fix the issue of encoding time fields with incorrect timezone information #46033 @tangenta
  - Fix the issue that the Sort operator might cause TiDB to crash during the spill process #47538 @windtalker
  - Fix the issue that TiDB returns Can't find column for queries with GROUP_CONCAT #41957 @AilinKid
  - Fix the panic issue of batch-client in client-go #47691 @crazycs520
  - Fix the issue of incorrect memory usage estimation in INDEX_LOOKUP_HASH_JOIN #47788 @SeaRise

  - Fix the issue of uneven workload caused by the rejoining of a TiFlash node that has been offline for a long time #35418 @windtalker
  - Fix the issue that the chunk cannot be reused when the HashJoin operator performs probe #48082 @wshwsh12
  - Fix the issue that the COALESCE() function returns incorrect result type for DATE type parameters #46475 @xzhangxian1008
  - Fix the issue that UPDATE statements with subqueries are incorrectly converted to PointGet #48171 @hi-rustin
  - Fix the issue that incorrect results are returned when the cached execution plans contain the comparison between date types and unix_timestamp #48165 @qw4990
  - Fix the issue that an error is reported when default inline common table expressions (CTEs) with aggregate functions or window functions are referenced by recursive CTEs #47881 @elsa0520
  - Fix the issue that the optimizer mistakenly selects IndexFullScan to reduce sort introduced by window functions #46177 @qw4990
  - Fix the issue that multiple references to CTEs result in incorrect results due to condition pushdown of CTEs #47881 @winoros
  - Fix the issue that the MySQL compression protocol cannot handle large loads of data (>=16M) #47152 #47157 #47161 @dveeden
  - Fix the issue that TiDB does not read cgroup resource limits when it is started with systemd #47442 @hawkingrei
- TiKV
  - Fix the issue that retrying prewrite requests in the pessimistic transaction mode might cause the risk of data inconsistency in rare cases #11187 @MyonKeminta
- PD
  - Fix the issue that evict-leader-scheduler might lose configuration #6897 @HuSharp
  - Fix the issue that after a store goes offline, the monitoring metric of its statistics is not deleted #7180 @rleungx

  - Fix the issue that canSync and hasMajority might be calculated incorrectly for clusters adopting the Data Replication Auto Synchronous (DR Auto-Sync) mode when the configuration of Placement Rules is complex #7201 @disksing
  - Fix the issue that the rule checker does not add Learners according to the configuration of Placement Rules #7185 @nolouch
  - Fix the issue that TiDB Dashboard cannot read PD trace data correctly #7253 @nolouch
  - Fix the issue that PD might panic due to empty Regions obtained internally #7261 @lhy1024
  - Fix the issue that available_stores is calculated incorrectly for clusters adopting the Data Replication Auto Synchronous (DR Auto-Sync) mode #7221 @disksing
  - Fix the issue that PD might delete normal Peers when TiKV nodes are unavailable #7249 @lhy1024
  - Fix the issue that adding multiple TiKV nodes to a large cluster might cause TiKV heartbeat reporting to become slow or stuck #7248 @rleungx
- TiFlash
  - Fix the issue that the UPPER() and LOWER() functions return inconsistent results between TiDB and TiFlash #7695 @windtalker
  - Fix the issue that executing queries on empty partitions causes query failure #8220 @JaySon-Huang
  - Fix the panic issue caused by table creation failure when replicating TiFlash replicas #8217 @hongyunyan
- Tools
  - Backup & Restore (BR)
    - Fix the issue that PITR might skip restoring the CREATE INDEX DDL statement #47482 @Leavrth
    - Fix the issue that the log backup might get stuck in some scenarios when backing up large wide tables #15714 @YuJuncen
  - TiCDC

- Fix the performance issue caused by accessing NFS directories when replicating data to an object store sink #10041 @CharlesCheung96
- Fix the issue that the storage path is misspelled when claim-check is enabled #10036 @3AceShowHand
- Fix the issue that TiCDC scheduling is not balanced in some cases #9845 @3AceShowHand
- Fix the issue that TiCDC might get stuck when replicating data to Kafka #9855 @hicqu
- Fix the issue that the TiCDC processor might panic in some cases #9849 #9915 @hicqu @3AceShowHand
- Fix the issue that enabling kv-client.enable-multiplexing causes replication tasks to get stuck #9673 @fubinzh
- Fix the issue that an owner node gets stuck due to NFS failure when the redo log is enabled #9886 @3AceShowHand

1.7 Performance test

To learn about the performance of TiDB v7.5.0, you can refer to the TPC-C performance test report and Sysbench performance test report of the TiDB Dedicated cluster.

1.8 Contributors

We would like to thank the following contributors from the TiDB community:

- jgrande (First-time contributor)
- shawn0915

2 TiDB 7.4.0 Release Notes

Release date: October 12, 2023

TiDB version: 7.4.0

Quick access: Quick start | Installation packages

7.4.0 introduces the following key features and improvements:

Category	Feature	Description
	Improve the performance and	Before v7.4.0, tasks such as ADD INDEX or

Category	Feature	Description
Reliability and Availability	stability of IMPORT INTO and ADD INDEX operations via global sort (experimental)	IMPORT INTO using the distributed execution framework meant localized and partial sorting, which ultimately led to TiKV doing a lot of extra work to make up for the partial sorting. These jobs also required TiDB nodes to allocate local disk space for sorting, before loading to TiKV. With the introduction of the Global Sorting feature in v7.4.0, data is temporarily stored in external shared storage (S3 in this version) for global sorting before being loaded into TiKV. This eliminates the need for TiKV to consume extra resources and significantly improves the performance and stability of operations like ADD INDEX and IMPORT INTO.
	Resource control for background tasks (experimental)	In v7.1.0, the Resource Control feature was introduced to mitigate resource and storage access interference between workloads. TiDB v7.4.0 applies this

Category	Feature	Description
		control to background tasks as well. In v7.4.0, Resource Control now identifies and manages the resources produced by background tasks, such as auto-analyze, Backup & Restore, bulk load with TiDB Lightning, and online DDL. This will eventually apply to all background tasks.
	TiFlash supports storage-computing separation and S3 (GA)	TiFlash disaggregated storage and compute architecture and S3 shared storage become generally available: • Disaggregates TiFlash's compute and storage, which is a milestone for elastic HTAP resource utilization. • Supports using S3-based storage engine, which can provide shared storage at a lower cost.
SQL	TiDB supports partition type management	Before v7.4.0, Range/List partitioned

Category	Feature	Description
		tables support partition management operations such as TRUNCATE, EXCHANGE, ADD, DROP, and REORGANIZE, and Hash/Key partitioned tables support partition management operations such as ADD and COALESCE. Now TiDB also supports the following partition type management operations: • Convert partitioned tables to non-partitioned tables • Partition existing non-partitioned tables • Modify partition types for existing tables
	MySQL 8.0 compatibility: support collation utf8mb4_0900_ai_ci	One notable change in MySQL 8.0 is that the default character set is utf8mb4, and the default collation of utf8mb4 is utf8mb4_0900_ai_ci. TiDB v7.4.0 adding support for this enhances compatibility with

Category	Feature	Description
		MySQL 8.0 so that migrations and replications from MySQL 8.0 databases with the default collation are now much smoother.
DB Operations and Observability	Specify the respective TiDB nodes to execute the IMPORT INTO and ADD INDEX SQL statements (experimental)	You have the flexibility to specify whether to execute IMPORT INTO or ADD INDEX SQL statements on some of the existing TiDB nodes or newly added TiDB nodes. This approach enables resource isolation from the rest of the TiDB nodes, preventing any impact on business operations while ensuring optimal performance for executing the preceding SQL statements.

2.1 Feature details

2.1.1 Scalability

- Support selecting the TiDB nodes to parallelly execute the backend ADD INDEX or IMPORT INTO tasks of the distributed execution framework (experimental) #46453 @ywqzzy

  Executing ADD INDEX or IMPORT INTO tasks in parallel in a resource-intensive cluster can consume a large amount of TiDB node resources, which can lead to cluster performance degradation. Starting from v7.4.0, you can use the system variable tidb_service_scope to control the service scope of each TiDB node under the TiDB Backend Task Distributed Execution Framework. You can select

several existing TiDB nodes or set the TiDB service scope for new TiDB nodes, and all parallel ADD INDEX and IMPORT INTO tasks only run on these nodes. This mechanism can avoid performance impact on existing services.

For more information, see documentation.

- Enhance the Partitioned Raft KV storage engine (experimental) #11515 #12842 @busyjay @tonyxuqqi @tabokie @bufferflies @5kbpers @SpadeA-Tang @nolouch

  TiDB v6.6.0 introduces the Partitioned Raft KV storage engine as an experimental feature, which uses multiple RocksDB instances to store TiKV Region data, and the data of each Region is independently stored in a separate RocksDB instance.

  In v7.4.0, TiDB further improves the compatibility and stability of the Partitioned Raft KV storage engine. Through large-scale data testing, the compatibility with TiDB ecosystem tools and features such as DM, Dumpling, TiDB Lightning, TiCDC, BR, and PITR is ensured. Additionally, the Partitioned Raft KV storage engine provides more stable performance under mixed read and write workloads, making it especially suitable for write-heavy scenarios. Furthermore, each TiKV node now supports 8 core CPUs and can be configured with 8 TB data storage, and 64 GB memory.

  For more information, see documentation.

- TiFlash supports the disaggregated storage and compute architecture (GA) #6882 @JaySon-Huang @JinheLin @breezewish @lidezhu @CalvinNeo @Lloyd-Pottiger

  In v7.0.0, TiFlash introduces the disaggregated storage and compute architecture as an experimental feature. With a series of improvements, the disaggregated storage and compute architecture for TiFlash becomes GA starting from v7.4.0.

  In this architecture, TiFlash nodes are divided into two types (Compute Nodes and Write Nodes) and support object storage that is compatible with S3 API. Both types of nodes can be independently scaled for computing or storage capacities. In the disaggregated storage and compute architecture, you can use TiFlash in the same

way as the coupled storage and compute architecture, such as creating TiFlash replicas, querying data, and specifying optimizer hints.

Note that the TiFlash **disaggregated storage and compute architecture** and **coupled storage and compute architecture** cannot be used in the same cluster or converted to each other. You can configure which architecture to use when you deploy TiFlash.

For more information, see documentation.

2.1.2 Performance

- Support pushing down the JSON operator MEMBER OF to TiKV #46307 @wshwsh12
  - value MEMBER OF(json_array)

  For more information, see documentation.
- Support pushing down window functions with any frame definition type to TiFlash #7376 @xzhangxian1008

  Before v7.4.0, TiFlash does not support window functions containing PRECEDING or FOLLOWING, and all window functions containing such frame definitions cannot be pushed down to TiFlash. Starting from v7.4.0, TiFlash supports frame definitions of all window functions. This feature is enabled automatically, and window functions containing frame definitions will be automatically pushed down to TiFlash for execution when the related requirements are met.
- Introduce cloud storage-based global sort capability to improve the performance and stability of ADD INDEX and IMPORT INTO tasks in parallel execution (experimental) #45719 @wjhuang2016

  Before v7.4.0, when executing tasks like ADD INDEX or IMPORT INTO in the distributed parallel execution framework, each TiDB node needs to allocate a significant amount of local disk space for sorting encoded index KV pairs and table data KV pairs. However, due to the lack of global sorting capability, there might be overlapping data between different TiDB nodes and within each individual node during the process. As a result, TiKV has to constantly perform compaction operations while importing these KV pairs into its storage engine,

which impacts the performance and stability of ADD INDEX and IMPORT INTO.

In v7.4.0, TiDB introduces the Global Sort feature. Instead of writing the encoded data locally and sorting it there, the data is now written to cloud storage for global sorting. Once sorted, both the indexed data and table data are imported into TiKV in parallel, thereby improving performance and stability.

For more information, see documentation.

- Support caching execution plans for non-prepared statements (GA) #36598 @qw4990

  TiDB v7.0.0 introduces non-prepared plan cache as an experimental feature to improve the load capacity of concurrent OLTP. In v7.4.0, this feature becomes GA. The execution plan cache will be applied to more scenarios, thereby improving the concurrent processing capacity of TiDB.

  Enabling the non-prepared plan cache might incur additional memory and CPU overhead and might not be suitable for all situations. Starting from v7.4.0, this feature is disabled by default. You can enable it using tidb_enable_non_prepared_plan_cache and control the cache size using tidb_session_plan_cache_size.

  Additionally, this feature does not support DML statements by default and has certain restrictions on SQL statements. For more details, see Restrictions.

  For more information, see documentation.

2.1.3 Reliability

- TiFlash supports query-level data spilling #7738 @windtalker

  Starting from v7.0.0, TiFlash supports controlling data spilling for three operators: GROUP BY, ORDER BY, and JOIN. This feature prevents issues such as query termination or system crashes when the data size exceeds the available memory. However, managing spilling for each operator individually can be cumbersome and ineffective for overall resource control.

In v7.4.0, TiFlash introduces the query-level data spilling. By setting the memory limit for a query on a TiFlash node using tiflash_mem_quota_query_per_node and the memory ratio that triggers data spilling using tiflash_query_spill_ratio, you can conveniently manage the memory usage of a query and have better control over TiFlash memory resources.

For more information, see documentation.

- Support user-defined TiKV read timeout #45380 @crazycs520

  Normally, TiKV processes requests very quickly, in a matter of milliseconds. However, when a TiKV node encounters disk I/O jitter or network latency, the request processing time can increase significantly. In versions earlier than v7.4.0, the timeout limit for TiKV requests is fixed and unadjustable. Hence, TiDB has to wait for a fixed-duration timeout response when a TiKV node encounters issues, which results in a noticeable impact on application query performance during jitter.

  TiDB v7.4.0 introduces a new system variable tikv_client_read_timeout, which lets you customize the timeout for RPC read requests that TiDB sends to TiKV in a query. It means that when the request sent to a TiKV node is delayed due to disk or network issues, TiDB can time out faster and resend the request to other TiKV nodes, thus reducing query latency. If timeouts occur for all TiKV nodes, TiDB will retry using the default timeout. Additionally, you can also use the optimizer hint /*+ SET_VAR(TIKV_CLIENT_READ_TIMEOUT=N) */ in a query to set the timeout for TiDB to send a TiKV RPC read request. This enhancement gives TiDB the flexibility to adapt to unstable network or storage environments, improving query performance and enhancing the user experience.

  For more information, see documentation.

- Support temporarily modifying some system variable values using an optimizer hint #45892 @winoros

  TiDB v7.4.0 introduces the optimizer hint SET_VAR(), which is similar to that of MySQL 8.0. By including the hint SET_VAR() in SQL statements, you can temporarily modify the value of system variables during statement execution. This helps you set the environment for different

statements. For example, you can actively increase the parallelism of resource-intensive SQL statements or change the optimizer behavior through variables.

You can find the system variables that can be modified using the hint SET_VAR() in system variables. It is strongly recommended not to modify variables that are not explicitly supported, as this might cause unpredictable behavior.

For more information, see documentation.

- TiFlash supports resource control #7660 @guo-shaoge

  In TiDB v7.1.0, the resource control feature becomes generally available and provides resource management capabilities for TiDB and TiKV. In v7.4.0, TiFlash supports the resource control feature, improving the overall resource management capabilities of TiDB. The resource control of TiFlash is fully compatible with the existing TiDB resource control feature, and the existing resource groups will manage the resources of TiDB, TiKV, and TiFlash at the same time.

  To control whether to enable the TiFlash resource control feature, you can configure the TiFlash parameter enable_resource_control. After enabling this feature, TiFlash performs resource scheduling and management based on the resource group configuration of TiDB, ensuring the reasonable allocation and use of overall resources.

  For more information, see documentation.

- TiFlash supports the pipeline execution model (GA) #6518 @SeaRise

  Starting from v7.2.0, TiFlash introduces a pipeline execution model. This model centrally manages all thread resources and schedules task execution uniformly, maximizing the utilization of thread resources while avoiding resource overuse. In v7.4.0, TiFlash improves the statistics of thread resource usage, and the pipeline execution model becomes a GA feature and is enabled by default. Since this feature is mutually dependent with the TiFlash resource control feature, TiDB v7.4.0 removes the variable tidb_enable_tiflash_pipeline_model used to control whether to enable the pipeline execution model in previous versions. Instead, you can enable or disable the pipeline execution

model and the TiFlash resource control feature by configuring the TiFlash parameter tidb_enable_resource_control.

For more information, see documentation.

- Add the option of optimizer mode #46080 @time-and-fate

  In v7.4.0, TiDB introduces a new system variable tidb_opt_objective, which controls the estimation method used by the optimizer. The default value moderate maintains the previous behavior of the optimizer, where it uses runtime statistics to adjust estimations based on data changes. If this variable is set to determinate, the optimizer generates execution plans solely based on statistics without considering runtime corrections.

  For long-term stable OLTP applications or situations where you are confident in the existing execution plans, it is recommended to switch to determinate mode after testing. This reduces potential plan changes.

  For more information, see documentation.

- TiDB resource control supports managing background tasks (experimental) #44517 @glorv

  Background tasks, such as data backup and automatic statistics collection, are low-priority but consume many resources. These tasks are usually triggered periodically or irregularly. During execution, they consume a lot of resources, thus affecting the performance of online high-priority tasks. Starting from v7.4.0, the TiDB resource control feature supports managing background tasks. This feature reduces the performance impact of low-priority tasks on online applications, enabling rational resource allocation, and greatly improving cluster stability.

  TiDB supports the following types of background tasks:

  - lightning: perform import tasks using TiDB Lightning or IMPORT INTO.
  - br: perform backup and restore tasks using BR. PITR is not supported.

  - ddl: control the resource usage during the batch data write back phase of Reorg DDLs.
  - stats: the collect statistics tasks that are manually executed or automatically triggered by TiDB.

  By default, the task types that are marked as background tasks are empty, and the management of background tasks is disabled. This default behavior is the same as that of versions prior to TiDB v7.4.0. To manage background tasks, you need to manually modify the background task types of the default resource group.

  For more information, see documentation.

- Enhance the ability to lock statistics #46351 @hi-rustin

  In v7.4.0, TiDB has enhanced the ability to lock statistics. Now, to ensure operational security, locking and unlocking statistics require the same privileges as collecting statistics. In addition, TiDB supports locking and unlocking statistics for specific partitions, providing greater flexibility. If you are confident in queries and execution plans in the database and want to prevent any changes from occurring, you can lock statistics to enhance stability.

  For more information, see documentation.

- Introduce a system variable to control whether to select hash joins for tables #46695 @coderplay

  MySQL 8.0 introduces hash joins for tables as a new feature. This feature is primarily used to join two relatively large tables and result sets. However, for transactional workloads, or some applications running on MySQL 5.7, hash joins for tables might pose a performance risk. MySQL provides the optimizer_switch to control whether to select hash joins at the global or session level.

  Starting from v7.4.0, TiDB introduces the system variable tidb_opt_enable_hash_join to have control over hash joins for tables. It is enabled by default (ON). If you are sure that you do not need to select hash joins between tables in your execution plan, you can modify the variable to OFF to reduce the possibility of execution plan rollbacks and improve system stability.

  For more information, see documentation.

2.1.4 SQL

- TiDB supports partition type management #42728 @mjonss

  Before v7.4.0, partition types of partitioned tables in TiDB cannot be modified. Starting from v7.4.0, TiDB supports modifying partitioned tables to non-partitioned tables or non-partitioned tables to partitioned tables, and supports changing partition types. Hence, now you can flexibly adjust the partition type and number for a partitioned table. For example, you can use the ALTER TABLE t PARTITION BY ... statement to modify the partition type.

  For more information, see documentation.

- TiDB supports using the ROLLUP modifier and the GROUPING function #44487 @AilinKid

  The WITH ROLLUP modifier and GROUPING function are commonly used in data analysis for multi-dimensional data summarization. Starting from v7.4.0, you can use the WITH ROLLUP modifier and GROUPING function in the GROUP BY clause. For example, you can use the WITH ROLLUP modifier in the SELECT ... FROM ... GROUP BY ... WITH ROLLUP syntax.

  For more information, see documentation.

2.1.5 DB operations

- Support collation utf8mb4_0900_ai_ci and utf8mb4_0900_bin #37566 @YangKeao @zimulala @bb7133

  TiDB v7.4.0 enhances the support for migrating data from MySQL 8.0 and adds two collations: utf8mb4_0900_ai_ci and utf8mb4_0900_bin. utf8mb4_0900_ai_ci is the default collation in MySQL 8.0.

  TiDB v7.4.0 also introduces the system variable default_collation_for_utf8mb4 which is compatible with MySQL 8.0. This enables you to specify the default collation for the utf8mb4 character set and provides compatibility with migration or data replication from MySQL 5.7 or earlier versions.

  For more information, see documentation.

2.1.6 Observability

- Support adding session connection IDs and session aliases to logs #46071 @lcwangchao

  When you troubleshoot a SQL execution problem, it is often necessary to correlate the contents of TiDB component logs to pinpoint the root cause. Starting from v7.4.0, TiDB can write session connection IDs (CONNECTION_ID) to session-related logs, including TiDB logs, slow query logs, and slow logs from the coprocessor on TiKV. You can correlate the contents of several types of logs based on session connection IDs to improve troubleshooting and diagnostic efficiency.

  In addition, by setting the session-level system variable tidb_session_alias, you can add custom identifiers to the logs mentioned above. With this ability to inject your application identification information into the logs, you can correlate the contents of the logs with the application, build the link from the application to the logs, and reduce the difficulty of diagnosis.

- TiDB Dashboard supports displaying execution plans in a table view #1589 @baurine

  In v7.4.0, TiDB Dashboard supports displaying execution plans on the **Slow Query** and **SQL Statement** pages in a table view to improve the diagnosis experience.

  For more information, see documentation.

2.1.7 Data migration

- Enhance the IMPORT INTO feature #46704 @D3Hunter

  Starting from v7.4.0, you can add the CLOUD_STORAGE_URI option in the IMPORT INTO statement to enable the global sorting feature (experimental), which helps boost import performance and stability. In the CLOUD_STORAGE_URI option, you can specify a cloud storage address for the encoded data.

  In addition, in v7.4.0, the IMPORT INTO feature introduces the following functionalities:

  - Support configuring the Split_File option, which allows you to split a large CSV file into multiple 256 MiB small CSV files for parallel processing, improving import performance.
  - Support importing compressed CSV and SQL files. The supported compression formats include .gzip, .gz, .zstd, .zst, and .snappy.

  For more information, see documentation.

- Dumpling supports the user-defined terminator when exporting data to CSV files #46982 @GMHDBJD

  Before v7.4.0, Dumpling uses "\r\n" as the line terminator when exporting data to a CSV file. As a result, certain downstream systems that only recognize "\n" as the terminator cannot parse the exported CSV file, or have to use a third-party tool for conversion before parsing the file.

  Starting from v7.4.0, Dumpling introduces a new parameter --csv-line-terminator. This parameter allows you to specify a desired terminator when you export data to a CSV file. This parameter supports "\r\n" and "\n". The default terminator is "\r\n" to keep consistent with earlier versions.

  For more information, see documentation.

- TiCDC supports replicating data to Pulsar #9413 @yumchina @asddongmen

  Pulsar is a cloud-native and distributed message streaming platform that significantly enhances your real-time data streaming experience. Starting from v7.4.0, TiCDC supports replicating change data to Pulsar in canal-json format to achieve seamless integration with Pulsar. With this feature, TiCDC provides you with the ability to easily capture and replicate TiDB changes to Pulsar, offering new possibilities for data processing and analytics capabilities. You can develop your own consumer applications that read and process newly generated change data from Pulsar to meet specific business needs.

  For more information, see documentation.

- TiCDC improves large message handling with claim-check pattern #9153 @3AceShowHand

Before v7.4.0, TiCDC is unable to send large messages exceeding the maximum message size (max.message.bytes) of Kafka to downstream. Starting from v7.4.0, when configuring a changefeed with Kafka as the downstream, you can specify an external storage location for storing the large message, and send a reference message containing the address of the large message in the external storage to Kafka. When consumers receive this reference message, they can retrieve the message content from the external storage address.

For more information, see documentation.

2.2 Compatibility changes

Note:

This section provides compatibility changes you need to know when you upgrade from v7.3.0 to the current version (v7.4.0). If you are upgrading from v7.2.0 or earlier versions to the current version, you might also need to check the compatibility changes introduced in intermediate versions.

2.2.1 Behavior changes

- Starting with v7.4.0, TiDB is compatible with essential features of MySQL 8.0, and version() returns the version prefixed with 8.0.11.
- After TiFlash is upgraded to v7.4.0 from an earlier version, in-place downgrading to the original version is not supported. This is because, starting from v7.4, TiFlash optimizes the data compaction logic of PageStorage V3 to reduce the read and write amplification generated during data compaction, which leads to changes to some of the underlying storage file names.
- A TIDB_PARSE_TSO_LOGICAL() function is added to allow the extraction of the logical part of the TSO timestamp.
- The information_schema.CHECK_CONSTRAINTS table is added for improved compatibility with MySQL 8.0.

2.2.2 System variables

Variable name	Change type	Description
tidb_enable_tiflash_pipeline_model	Deleted	This variable was used to control whether to enable the TiFlash pipeline execution model. Starting from v7.4.0, the TiFlash pipeline execution model is automatically enabled when the TiFlash resource control feature is enabled.
tidb_enable_non_prepared_plan_cache	Modified	Changes the default value from ON to OFF after further tests, meaning that non-prepared execution plan cache is disabled.
default_collation_for_utf8mb4	Newly added	Controls the default collation for the utf8mb4 character set. The default value is utf8mb4_bin.
tidb_cloud_storage_uri	Newly added	Specifies the cloud storage URI to enable Global Sort.
tidb_opt_enable_hash_join	Newly added	Controls whether the optimizer will select hash joins for tables. The value is ON by default. If set to OFF, the optimizer avoids selecting a hash join of a table unless there is no other execution plan available.
tidb_opt_objective	Newly added	This variable controls the objective of the optimizer. moderate maintains the default behavior in versions prior to TiDB v7.4.0, where the optimizer tries to use more information to generate better execution plans. determinate mode tends to be more conservative and makes the execution plan more stable.
tidb_schema_version_cache_limit	Newly added	This variable limits how many historical schema versions can be cached in a TiDB instance. The default value is 16, which means that TiDB caches 16 historical schema versions by default.
tidb_service_scope	Newly added	This variable is an instance-level system variable. You can use it to control the service scope of TiDB nodes under the TiDB distributed execution framework. When you set tidb_service_scope of a TiDB node to background, the TiDB distributed execution framework schedules that TiDB node to execute background tasks, such as ADD INDEX and IMPORT INTO.
tidb_session_alias	Newly added	Controls the value of the session_alias column in the logs related to the current session.
tiflash_mem_quota_query_per_node	Newly added	Limits the maximum memory usage for a query on a TiFlash node. When the memory usage of a query exceeds this limit, TiFlash returns an error and

Variable name	Change type	Description
		terminates the query. The default value is 0, which means no limit.
tiflash_query_spill_ratio	Newly added	Controls the threshold for TiFlash query-level spilling. The default value is 0.7.
tikv_client_read_timeout	Newly added	Controls the timeout for TiDB to send a TiKV RPC read request in a query. The default value 0 indicates that the default timeout (usually 40 seconds) is used.

2.2.3 Configuration file parameters

Configuration file	Configuration parameter	Change type	Description
TiDB	enable-stats-cache-mem-quota	Modified	The default value is changed from false to true, which means the memory limit for caching TiDB statistics is enabled by default.
TiKV	rocksdb.[defaultcf|writecf|lockcf].periodic-compaction-seconds	Modified	The default value is changed from "30d" to "0s" to disable periodic compaction of RocksDB by default. This change avoids a significant number of compactions being triggered after the TiDB upgrade, which affects the read and write performance of the frontend.
TiKV	rocksdb.[defaultcf|writecf|lockcf].ttl	Modified	The default value is changed from "30d" to "0s" so that SST files do not trigger compactions by default due to TTL, which avoids affecting the read and write performance of the frontend.
TiFlash	flash.compact_log_min_gap	Newly added	When the gap between the applied_index advanced by the current Raft state machine and the applied_index at the last disk spilling exceeds compact_log_min_gap, TiFlash executes the CompactLog command from TiKV and spills data to disk.
TiFlash	profiles.default.enable_resource_control	Newly added	Controls whether to enable the TiFlash resource control feature.
TiFlash	storage.format_version	Modified	Change the default value from 4 to 5. The new format can reduce the number

Configuration file	Configuration parameter	Change type	Description
			of physical files by merging smaller files.
Dumpling	--csv-line-terminator	Newly added	Specifies the desired terminator of CSV files . This option supports "\r\n" and "\n". The default value is "\r\n", which is consistent with the earlier versions.
TiCDC	claim-check-storage-uri	Newly added	When large-message-handle-option is set to claim-check, claim-check-storage-uri must be set to a valid external storage address. Otherwise, creating a changefeed results in an error.
TiCDC	large-message-handle-compression	Newly added	Controls whether to enable compression during encoding. The default value is empty, which means not enabled.
TiCDC	large-message-handle-option	Modified	This configuration item adds a new value claim-check. When it is set to claim-check, TiCDC Kafka sink supports sending the message to external storage when the message size exceeds the limit and sends a message to Kafka containing the address of this large message in external storage.

2.3 Deprecated features

- Mydumper will be deprecated in v7.5.0 and most of its features have been replaced by Dumpling. It is strongly recommended that you use Dumpling instead of mydumper.
- TiKV-importer will be deprecated in v7.5.0. It is strongly recommended that you use the Physical Import Mode of TiDB Lightning as an alternative.

2.4 Improvements

- TiDB
  - Optimize memory usage and performance for ANALYZE operations on partitioned tables #47071 #47104 #46804 @hawkingrei

  - Optimize memory usage and performance for statistics garbage collection #31778 @winoros
  - Optimize the pushdown of `limit` for index merge intersections to improve query performance #46863 @AilinKid
  - Improve the cost model to minimize the chances of mistakenly choosing a full table scan when `IndexLookup` involves many table retrieval tasks #45132 @qw4990
  - Optimize the join elimination rule to improve the query performance of `join` on unique keys #46248 @fixdb
  - Change the collation of multi-valued index columns to `binary` to avoid execution failure #46717 @YangKeao
- TiKV
  - Optimize memory usage of Resolver to prevent OOM #15458 @overvenus
  - Eliminate LRUCache in Router objects to reduce memory usage and prevent OOM #15430 @Connor1996
  - Reduce memory usage of TiCDC Resolver #15412 @overvenus
  - Reduce memory fluctuations caused by RocksDB compaction #15324 @overvenus
  - Reduce memory consumption in the flow control module of Partitioned Raft KV #15269 @overvenus
  - Add the backoff mechanism for the PD client in the process of connection retries, which gradually increases retry intervals during error retries to reduce PD pressure #15428 @nolouch
  - Support dynamically adjusting `background_compaction` of RocksDB #15424 @glorv
- PD
  - Optimize TSO tracing information for easier investigation of TSO-related issues #6856 @tiancaiamao
  - Support reusing HTTP Client connections to reduce memory usage #6913 @nolouch
  - Improve the speed of PD automatically updating cluster status when the backup cluster is disconnected #6883 @disksing
  - Enhance the configuration retrieval method of the resource control client to dynamically fetch the latest configurations #7043 @nolouch

- TiFlash
  - Improve write performance during random write workloads by optimizing the spilling policy of the TiFlash write process #7564 @CalvinNeo
  - Add more metrics about the Raft replication process for TiFlash #8068 @CalvinNeo
  - Reduce the number of small files to avoid potential exhaustion of file system inodes #7595 @hongyunyan
- Tools
  - Backup & Restore (BR)
    - Alleviate the issue that the latency of the PITR log backup progress increases when Region leadership migration occurs #13638 @YuJuncen
    - Enhance support for connection reuse of log backup and PITR restore tasks by setting MaxIdleConns and MaxIdleConnsPerHost parameters in the HTTP client #46011 @Leavrth
    - Improve fault tolerance of BR when it fails to connect to PD or external S3 storage #42909 @Leavrth
    - Add a new restore parameter WaitTiflashReady. When this parameter is enabled, the restore operation will be completed after TiFlash replicas are successfully replicated #43828 #46302 @3pointer
    - Reduce the CPU overhead of log backup resolve lock #40759 @3pointer
  - TiCDC
    - Optimize the execution logic of replicating the ADD INDEX DDL operations to avoid blocking subsequent DML statements #9644 @sdojjy
  - TiDB Lightning
    - Optimize the retry logic of TiDB Lightning during the Region scatter phase #46203 @mittalrishabh
    - Optimize the retry logic of TiDB Lightning for the no leader error during the data import phase #46253 @lance6716

2.5 Bug fixes

- TiDB
  - Fix the issue that the BatchPointGet operator returns incorrect results for tables that are not hash partitioned #45889 @Defined2014
  - Fix the issue that the BatchPointGet operator returns incorrect results for hash partitioned tables #46779 @jiyfhust
  - Fix the issue that the TiDB parser remains in a state and causes parsing failure #45898 @qw4990
  - Fix the issue that EXCHANGE PARTITION does not check constraints #45922 @mjonss
  - Fix the issue that the tidb_enforce_mpp system variable cannot be correctly restored #46214 @djshow832
  - Fix the issue that the _ in the LIKE clause is incorrectly handled #46287 #46618 @Defined2014
  - Fix the issue that the schemaTs is set to 0 when TiDB fails to obtain the schema #46325 @hihihuhu
  - Fix the issue that Duplicate entry might occur when AUTO_ID_CACHE=1 is set #46444 @tiancaiamao
  - Fix the issue that TiDB recovers slowly after a panic when AUTO_ID_CACHE=1 is set #46454 @tiancaiamao
  - Fix the issue that the next_row_id in SHOW CREATE TABLE is incorrect when AUTO_ID_CACHE=1 is set #46545 @tiancaiamao
  - Fix the panic issue that occurs during parsing when using CTE in subqueries #45838 @djshow832
  - Fix the issue that restrictions on partitioned tables remain on the original table when EXCHANGE PARTITION fails or is canceled #45920 #45791 @mjonss
  - Fix the issue that the definition of List partitions does not support using both NULL and empty strings #45694 @mjonss
  - Fix the issue of not being able to detect data that does not comply with partition definitions during partition exchange #46492 @mjonss
  - Fix the issue that the tmp-storage-quota configuration does not take effect #45161 #26806 @wshwsh12

- Fix the issue that the WEIGHT_STRING() function does not match the collation #45725 @dveeden
- Fix the issue that an error in Index Join might cause the query to get stuck #45716 @wshwsh12
- Fix the issue that the behavior is inconsistent with MySQL when comparing a DATETIME or TIMESTAMP column with a number constant #38361 @yibin87
- Fix the incorrect result that occurs when comparing unsigned types with Duration type constants #45410 @wshwsh12
- Fix the issue that access path pruning logic ignores the READ_FROM_STORAGE(TIFLASH[...]) hint, which causes the Can't find a proper physical plan error #40146 @AilinKid
- Fix the issue that GROUP_CONCAT cannot parse the ORDER BY column #41986 @AilinKid
- Fix the issue that HashCode is repeatedly calculated for deeply nested expressions, which causes high memory usage and OOM #42788 @AilinKid
- Fix the issue that the cast(col)=range condition causes FullScan when CAST has no precision loss #45199 @AilinKid
- Fix the issue that when Aggregation is pushed down through Union in MPP execution plans, the results are incorrect #45850 @AilinKid
- Fix the issue that bindings with in (?) cannot match in (?, ... ?) #44298 @qw4990
- Fix the error caused by not considering the connection collation when non-prep plan cache reuses the execution plan #47008 @qw4990
- Fix the issue that no warning is reported when an executed plan does not hit the plan cache #46159 @qw4990
- Fix the issue that plan replayer dump explain reports an error #46197 @time-and-fate
- Fix the issue that executing DML statements with CTE can cause panic #46083 @winoros
- Fix the issue that the TIDB_INLJ hint does not take effect when joining two sub-queries #46160 @qw4990
- Fix the issue that the results of MERGE_JOIN are incorrect #46580 @qw4990

- TiKV
  - Fix the issue that TiKV fails to start when Titan is enabled and the `Blob file deleted twice` error occurs #15454 @Connor1996
  - Fix the issue of no data in the Thread Voluntary and Thread Nonvoluntary monitoring panels #15413 @SpadeA-Tang
  - Fix the data error of continuously increasing raftstore-applys #15371 @Connor1996
  - Fix the TiKV panic issue caused by incorrect metadata of Region #13311 @zyguan
  - Fix the issue of QPS dropping to 0 after switching from `sync_recovery` to `sync` #15366 @nolouch
  - Fix the issue that Online Unsafe Recovery does not abort on timeout #15346 @Connor1996
  - Fix the potential memory leak issue caused by CpuRecord #15304 @overvenus
  - Fix the issue that `"Error 9002: TiKV server timeout"` occurs when the backup cluster is down and the primary cluster is queried #12914 @Connor1996
  - Fix the issue that the backup TiKV gets stuck when TiKV restarts after the primary cluster recovers #12320 @disksing
- PD
  - Fix the issue that the Region information is not updated and saved during Flashback #6912 @overvenus
  - Fix the issue of slow switching of PD Leaders due to slow synchronization of store config #6918 @bufferflies
  - Fix the issue that the groups are not considered in Scatter Peers #6962 @bufferflies
  - Fix the issue that RU consumption less than 0 causes PD to crash #6973 @CabinfeverB
  - Fix the issue that modified isolation levels are not synchronized to the default placement rules #7121 @rleungx
  - Fix the issue that the client-go regularly updating `min-resolved-ts` might cause PD OOM when the cluster is large #46664 @HuSharp
- TiFlash

  - Fix the issue that the max_snapshot_lifetime metric is displayed incorrectly on Grafana #7713 @JaySon-Huang
  - Fix the issue that some metrics about the maximum duration are not correct #8076 @CalvinNeo
  - Fix the issue that TiDB incorrectly reports that an MPP task has failed #7177 @yibin87
- Tools
  - Backup & Restore (BR)
    - Fix an issue that the misleading error message resolve lock timeout covers up the actual error when backup fails #43236 @YuJuncen
    - Fix the issue that recovering implicit primary keys using PITR might cause conflicts #46520 @3pointer
    - Fix the issue that recovering meta-kv using PITR might cause errors #46578 @Leavrth
    - Fix the errors in BR integration test cases #45561 @purelind
  - TiCDC
    - Fix the issue that TiCDC accesses the invalid old address during PD scaling up and down #9584 @fubinzh @asddongmen
    - Fix the issue that changefeed fails in some scenarios #9309 #9450 #9542 #9685 @hicqu @CharlesCheung96
    - Fix the issue that replication write conflicts might occur when the unique keys for multiple rows are modified in one transaction on the upstream #9430 @sdojjy
    - Fix the issue that a replication error occurs when multiple tables are renamed in the same DDL statement on the upstream #9476 #9488 @CharlesCheung96 @asddongmen
    - Fix the issue that Chinese characters are not validated in CSV files #9609 @CharlesCheung96
    - Fix the issue that upstream TiDB GC is blocked after all changefeeds are removed #9633 @sdojjy
    - Fix the issue of uneven distribution of write keys among nodes when scale-out is enabled #9665 @sdojjy

        - Fix the issue that sensitive user information is recorded in the logs #9690 @sdojjy
    - TiDB Data Migration (DM)
        - Fix the issue that DM cannot handle conflicts correctly with case-insensitive collations #9489 @hihihuhu
        - Fix the DM validator deadlock issue and enhance retries #9257 @D3Hunter
        - Fix the issue that replication lag returned by DM keeps growing when a failed DDL is skipped and no subsequent DDLs are executed #9605 @D3Hunter
        - Fix the issue that DM cannot properly track upstream table schemas when skipping online DDLs #9587 @GMHDBJD
        - Fix the issue that DM skips all DMLs when resuming a task in optimistic mode #9588 @GMHDBJD
        - Fix the issue that DM skips partition DDLs in optimistic mode #9788 @GMHDBJD
    - TiDB Lightning
        - Fix the issue that inserting data returns an error after TiDB Lightning imports the NONCLUSTERED auto_increment and AUTO_ID_CACHE=1 tables #46100 @tiancaiamao
        - Fix the issue that checksum still reports errors when checksum = "optional" #45382 @lyzx2001
        - Fix the issue that data import fails when the PD cluster address changes #43436 @lichunzhu

2.6 Contributors

We would like to thank the following contributors from the TiDB community:

- aidendou
- coderplay
- fatelei
- highpon
- hihihuhu (First-time contributor)
- isabella0428
- jiyfhust

- JK1Zhang
- joker53-1 (First-time contributor)
- L-maple
- mittalrishabh
- paveyry
- shawn0915
- tedyu
- yumchina
- ZhuohaoHe

3 TiDB 7.3.0 Release Notes

Release date: August 14, 2023

TiDB version: 7.3.0

Quick access: Quick start | Installation packages

7.3.0 introduces the following major features. In addition to that, 7.3.0 also includes a series of enhancements (described in the Feature details section) to query stability in TiDB server and TiFlash. These enhancements are more miscellaneous in nature and not user-facing so they are not included in the following table.

Category	Feature	Description
Scalability and Performance	TiDB Lightning supports Partitioned Raft KV (experimental)	TiDB Lightning now supports the new Partitioned Raft KV architecture, as part of the near-term GA of the architecture.
Reliability and Availability	Add automatic conflict detection and resolution on data imports	The TiDB Lightning Physical Import Mode supports a new version of conflict detection, which implements the semantics of replacing (replace) or ignoring (ignore) conflict data when encountering conflicts. It automatically handles conflict data for you while improving the performance of conflict resolution.
	Manual management of	Queries might take longer than you expect. With the new watch list of

Category	Feature	Description
	runaway queries (experimental)	resource groups, you can now manage queries more effectively and either deprioritize or kill them. Allowing operators to mark target queries by exact SQL text, SQL digest, or plan digest and deal with the queries at a resource group level, this feature gives you much more control over the potential impact of unexpected large queries on a cluster.
SQL	Enhance operator control over query stability by adding more optimizer hints to the query planner	Added hints: NO_INDEX_JOIN(), NO_MERGE_JOIN(), NO_INDEX_MERGE_JOIN(), NO_HASH_JOIN(), NO_INDEX_HASH_JOIN()
DB Operations and Observability	Show the progress of statistics collection tasks	Support viewing the progress of ANALYZE tasks using the SHOW ANALYZE STATUS statement or through the mysql.analyze_jobs system table.

3.1 Feature details

3.1.1 Performance

- TiFlash supports the replica selection strategy #44106 @XuHuaiyu

  Before v7.3.0, TiFlash uses replicas from all its nodes for data scanning and MPP calculations to maximize performance. Starting from v7.3.0, TiFlash introduces the replica selection strategy and lets you configure it using the tiflash_replica_read system variable. This strategy supports selecting specific replicas based on the zone attributes of nodes and scheduling specific nodes for data scanning and MPP calculations.

  For a cluster that is deployed in multiple data centers and each data center has complete TiFlash data replicas, you can configure this strategy to only select TiFlash replicas from the current data center.

This means data scanning and MPP calculations are performed only on TiFlash nodes in the current data center, which avoids excessive network data transmission across data centers.

For more information, see documentation.

- TiFlash supports Runtime Filter within nodes #40220 @elsa0520

  Runtime Filter is a **dynamic predicate** generated during the query planning phase. In the process of table joining, these dynamic predicates can effectively filter out rows that do not meet the join conditions, reducing scan time and network overhead, and improving the efficiency of table joining. Starting from v7.3.0, TiFlash supports Runtime Filter within nodes, improving the overall performance of analytical queries. In some TPC-DS workloads, the performance can be improved by 10% to 50%.

  This feature is disabled by default in v7.3.0. To enable this feature, set the system variable tidb_runtime_filter_mode to LOCAL.

  For more information, see documentation.

- TiFlash supports executing common table expressions (CTEs) (experimental) #43333 @winoros

  Before v7.3.0, the MPP engine of TiFlash cannot execute queries that contain CTEs by default. To achieve the best execution performance within the MPP framework, you need to use the system variable tidb_opt_force_inline_cte to enforce inlining CTE.

  Starting from v7.3.0, TiFlash's MPP engine supports executing queries with CTEs without inlining them, allowing for optimal query execution within the MPP framework. In TPC-DS benchmark tests, compared with inlining CTEs, this feature has shown a 20% improvement in overall query execution speed for queries containing CTE.

  This feature is experimental and is disabled by default. It is controlled by the system variable tidb_opt_enable_mpp_shared_cte_execution.

3.1.2 Reliability

- Add new optimizer hints #45520 @qw4990

In v7.3.0, TiDB introduces several new optimizer hints to control the join methods between tables, including:

  - NO_MERGE_JOIN() selects join methods other than merge join.
  - NO_INDEX_JOIN() selects join methods other than index nested loop join.
  - NO_INDEX_MERGE_JOIN() selects join methods other than index nested loop merge join.
  - NO_HASH_JOIN() selects join methods other than hash join.
  - NO_INDEX_HASH_JOIN() selects join methods other than index nested loop hash join.

  For more information, see documentation.

- Manually mark queries that use resources more than expected (experimental) #43691 @Connor1996 @CabinfeverB

  In v7.2.0, TiDB automatically manages queries that use resources more than expected (Runaway Query) by automatically downgrading or canceling runaway queries. In actual practice, rules alone cannot cover all cases. Therefore, TiDB v7.3.0 introduces the ability to manually mark runaway queries. With the new command QUERY WATCH, you can mark runaway queries based on SQL text, SQL Digest, or execution plan, and the marked runaway queries can be downgraded or cancelled.

  This feature provides an effective intervention method for sudden performance issues in the database. For performance issues caused by queries, before identifying the root cause, this feature can quickly alleviate its impact on overall performance, thereby improving system service quality.

  For more information, see documentation.

3.1.3 SQL

- List and List COLUMNS partitioned tables support default partitions #20679 @mjonss @bb7133

  Before v7.3.0, when you use the INSERT statement to insert data into a List or List COLUMNS partitioned table, the data needs to meet the specified partitioning conditions of the table. If the data to be inserted

does not meet any of these conditions, either the execution of the statement will fail or the non-compliant data will be ignored.

Starting from v7.3.0, List and List COLUMNS partitioned tables support default partitions. After a default partition is created, if the data to be inserted does not meet any partitioning condition, it will be written to the default partition. This feature improves the usability of List and List COLUMNS partitioning, avoiding the execution failure of the INSERT statement or data being ignored due to data that does not meet partitioning conditions.

Note that this feature is a TiDB extension to MySQL syntax. For a partitioned table with a default partition, the data in the table cannot be directly replicated to MySQL.

For more information, see documentation.

3.1.4 Observability

- Show the progress of collecting statistics #44033 @hawkingrei

  Collecting statistics for large tables often takes a long time. In previous versions, you cannot see the progress of collecting statistics, and therefore cannot predict the completion time. TiDB v7.3.0 introduces a feature to show the progress of collecting statistics. You can view the overall workload, current progress, and estimated completion time for each subtask using the system table mysql.analyze_jobs or SHOW ANALYZE STATUS. In scenarios such as large-scale data import and SQL performance optimization, this feature helps you understand the overall task progress and improves the user experience.

  For more information, see documentation.

- Plan Replayer supports exporting historical statistics #45038 @time-and-fate

  Starting from v7.3.0, with the newly added dump with stats as of timestamp clause, you can use Plan Replayer to export the statistics of specified SQL-related objects at a specific point in time. During the diagnosis of execution plan issues, accurately capturing historical statistics can help analyze more precisely how the execution plan was generated at the time when the issue occurred. This helps identify the

root cause of the issue and greatly improves efficiency in diagnosing execution plan issues.

For more information, see documentation.

3.1.5 Data migration

- TiDB Lightning introduces a new version of conflict data detection and handling strategy #41629 @lance6716

  In previous versions, TiDB Lightning uses different conflict detection and handling methods for Logical Import Mode and Physical Import Mode, which are complex to configure and not easy for users to understand. In addition, Physical Import Mode cannot handle conflicts using the replace or ignore strategy. Starting from v7.3.0, TiDB Lightning introduces a unified conflict detection and handling strategy for both Logical Import Mode and Physical Import Mode. You can choose to report an error (error), replace (replace) or ignore (ignore) conflicting data when encountering conflicts. You can limit the number of conflict records, such as the task is interrupted and terminated after processing a specified number of conflict records. Furthermore, the system can record conflicting data for troubleshooting.

  For import data with many conflicts, it is recommended to use the new version of the conflict detection and handling strategy for better performance. In the lab environment, the new version strategy can improve the performance of conflict detection and handling up to three times faster than the old version. This performance value is for reference only. The actual performance might vary depending on your configuration, table structure, and the percentage of conflicting data. Note that the new version and the old version of the conflict strategy cannot be used at the same time. The old conflict detection and handling strategy will be deprecated in the future.

  For more information, see documentation.

- TiDB Lightning supports Partitioned Raft KV (experimental) #14916 @GMHDBJD

  TiDB Lightning now supports Partitioned Raft KV. This feature helps improve the data import performance of TiDB Lightning.

- TiDB Lightning introduces a new parameter enable-diagnose-log to enhance troubleshooting by printing more diagnostic logs #45497 @D3Hunter

  By default, this feature is disabled and TiDB Lightning only prints logs containing lightning/main. When enabled, TiDB Lightning prints logs for all packages (including client-go and tidb) to help diagnose issues related to client-go and tidb.

  For more information, see documentation.

3.2 Compatibility changes

Note:

This section provides compatibility changes you need to know when you upgrade from v7.2.0 to the current version (v7.3.0). If you are upgrading from v7.1.0 or earlier versions to the current version, you might also need to check the compatibility changes introduced in intermediate versions.

3.2.1 Behavior changes

- Backup & Restore (BR)
  - BR adds an empty cluster check before performing a full data restoration. By default, restoring data to a non-empty cluster is not allowed. If you want to force the restoration, you can use the --filter option to specify the corresponding table name to restore data to.
- TiDB Lightning
  - tikv-importer.on-duplicate is deprecated and replaced by conflict.strategy.
  - The max-error parameter, which controls the maximum number of non-fatal errors that TiDB Lightning can tolerate before stopping the migration task, no longer limits import data conflicts. The conflict.threshold parameter now controls the maximum number of conflicting records that can be tolerated.
- TiCDC
  - When Kafka sink uses Avro protocol, if the force-replicate parameter is set to true, TiCDC reports an error when creating a changefeed.

- Due to incompatibility between delete-only-output-handle-key-columns and force-replicate parameters, when both parameters are enabled, TiCDC reports an error when creating a changefeed.
- When the output protocol is Open Protocol, the UPDATE events only output the changed columns.

3.2.2 System variables

Variable name	Change type	Description
tidb_opt_enable_mpp_shared_cte_execution	Modified	This system variable takes effect starting from v7.3.0. It controls whether non-recursive Common Table Expressions (CTEs) can be executed in TiFlash MPP.
tidb_lock_unchanged_keys	Newly added	This variable is used to control in certain scenarios whether to lock the keys that are involved but not modified in a transaction.
tidb_opt_enable_non_eval_scalar_subquery	Newly added	Controls whether the EXPLAIN statement disables the execution of constant subqueries that can be expanded at the optimization stage.
tidb_skip_missing_partition_stats	Newly added	This variable controls the generation of GlobalStats when partition statistics are missing.
tiflash_replica_read	Newly added	Controls the strategy for selecting TiFlash replicas when a query requires the TiFlash engine.

3.2.3 Configuration file parameters

Configuration file	Configuration parameter	Change type	Description
TiDB	enable-32bits-connection-id	Newly added	Controls whether to enable the 32-bit connection ID feature.
TiDB	in-mem-slow-query-recent-num	Newly added	Controls the number of recently used slow queries that are cached in memory.
TiDB	in-mem-slow-query-topn-num	Newly added	Controls the number of slowest queries that are cached in memory.

Configuration file	Configuration parameter	Change type	Description
TiKV	coprocessor.region-bucket-size	Modified	Changes the default value from 96MiB to 50MiB.
TiKV	raft-engine.format-version	Modified	When using Partitioned Raft KV (storage.engine="partitioned-raft-kv"), Ribbon filter is used. Therefore, TiKV changes the default value from 2 to 5.
TiKV	raftdb.max-total-wal-size	Modified	When using Partitioned Raft KV (storage.engine="partitioned-raft-kv"), TiKV skips writing WAL. Therefore, TiKV changes the default value from "4GB" to 1, meaning that WAL is disabled.
TiKV	rocksdb.[defaultcf|writecf|lockcf].compaction-guard-min-output-file-size	Modified	Changes the default value from "1MB" to "8MB" to resolve the issue that compaction speed cannot keep up with the write speed during large data writes.
TiKV	rocksdb.[defaultcf|writecf|lockcf].format-version	Modified	When using Partitioned Raft KV (storage.engine="partitioned-raft-kv"), Ribbon filter is used. Therefore, TiKV changes the default value from 2 to 5.
TiKV	rocksdb.lockcf.write-buffer-size	Modified	When using Partitioned Raft KV (storage.engine="partitioned-raft-kv"), to speed up compaction on lockcf, TiKV changes the default value from "32MB" to "4MB".
TiKV	rocksdb.max-total-wal-size	Modified	When using Partitioned Raft KV (storage.engine="partitioned-raft-kv"), TiKV skips writing WAL. Therefore, TiKV changes the default value from "4GB" to 1, meaning that WAL is disabled.
TiKV	rocksdb.stats-dump-period	Modified	When using Partitioned Raft KV (storage.engine="partitioned-raft-kv"), to disable redundant log printing,

Configuration file	Configuration parameter	Change type	Description
			changes the default value from "10m" to "0".
TiKV	rocksdb.write-buffer-limit	Modified	To reduce the memory overhead of memtables, when storage.engine="raft-kv", TiKV changes the default value from 25% of the memory of the machine to 0, which means no limit. When using Partitioned Raft KV (storage.engine="partitioned-raft-kv"), TiKV changes the default value from 25% to 20% of the memory of the machine.
TiKV	storage.block-cache.capacity	Modified	When using Partitioned Raft KV (storage.engine="partitioned-raft-kv"), to compensate for the memory overhead of memtables, TiKV changes the default value from 45% to 30% of the size of total system memory.
TiFlash	storage.format_version	Modified	Introduces a new DTFile format format_version = 5 to reduce the number of physical files by merging smaller files. Note that this format is experimental and not enabled by default.
TiDB Lightning	tikv-importer.incremental-import	Deleted	TiDB Lightning parallel import parameter. Because it could easily be mistaken as an incremental import parameter, this parameter is now renamed to tikv-importer.parallel-import. If a user passes in the old parameter name, it will be automatically converted to the new one.
TiDB Lightning	tikv-importer.on-duplicate	Deprecated	Controls action to do when trying to insert a conflicting record in the logical import mode. Starting from v7.3.0, this parameter is replaced by conflict.strategy.
TiDB Lightning	conflict.max-record-rows	Newly added	The new version of strategy to handle conflicting data. It controls the maximum number of rows in the conflict_records table. The default value is 100.

Configuration file	Configuration parameter	Change type	Description
TiDB Lightning	conflict.strategy	Newly added	The new version of strategy to handle conflicting data. It includes the following options: "" (TiDB Lightning does not detect and process conflicting data), error (terminate the import and report an error if a primary or unique key conflict is detected in the imported data), replace (when encountering data with conflicting primary or unique keys, the new data is retained and the old data is overwritten.), ignore (when encountering data with conflicting primary or unique keys, the old data is retained and the new data is ignored.). The default value is "", that is, TiDB Lightning does not detect and process conflicting data.
TiDB Lightning	conflict.threshold	Newly added	Controls the upper limit of the conflicting data. When conflict.strategy="error", the default value is 0. When conflict.strategy="replace" or conflict.strategy="ignore", you can set it as a maxint.
TiDB Lightning	enable-diagnose-logs	Newly added	Controls whether to enable the diagnostic logs. The default value is false, that is, only the logs related to the import are output, and the logs of other dependent components are not output. When you set it to true, logs from both the import process and other dependent components are output, and GRPC debugging is enabled, which can be used for diagnosis.
TiDB Lightning	tikv-importer.parallel-import	Newly added	TiDB Lightning parallel import parameter. It replaces the existing tikv-importer.incremental-import parameter, which could be mistaken as an incremental import parameter and misused.
BR	azblob.encryption-scope	Newly added	BR provides encryption scope support for Azure Blob Storage.

Configuration file	Configuration parameter	Change type	Description
BR	azblob.encryption-key	Newly added	BR provides encryption key support for Azure Blob Storage.
TiCDC	large-message-handle-option	Newly added	Empty by default, which means that when the message size exceeds the limit of Kafka topic, the changefeed fails. When this configuration is set to "handle-key-only", if the message exceeds the size limit, only the handle key will be sent to reduce the message size; if the reduced message still exceeds the limit, then the changefeed fails.
TiCDC	sink.csv.binary-encoding-method	Newly added	The encoding method of binary data, which can be 'base64' or 'hex'. The default value is 'base64'.

3.2.4 System tables

- Add a new system table mysql.tidb_timers to store the metadata of internal timers.

3.3 Deprecated features

- TiDB
  - The Fast Analyze feature (experimental) for statistics will be deprecated in v7.5.0.
  - The incremental collection feature for statistics will be deprecated in v7.5.0.

3.4 Improvements

- TiDB
  - Introduce a new system variable tidb_opt_enable_non_eval_scalar_subquery to control whether the EXPLAIN statement executes subqueries in advance during the optimization phase #22076 @winoros
  - When Global Kill is enabled, you can terminate the current session by pressing Control+C #8854 @pingyu
  - Support the IS_FREE_LOCK() and IS_USED_LOCK() locking functions #44493 @dveeden
  - Optimize the performance of reading the dumped chunks from disk #45125 @YangKeao

  - Optimize the overestimation issue of the inner table of Index Join by using Optimizer Fix Controls #44855 @time-and-fate
- TiKV
  - Add the Max gap of safe-ts and Min safe ts region metrics and introduce the tikv-ctl get-region-read-progress command to better observe and diagnose the status of resolved-ts and safe-ts #15082 @ekexium
- PD
  - Support blocking the Swagger API by default when the Swagger server is not enabled #6786 @bufferflies
  - Improve the high availability of etcd #6554 #6442 @lhy1024
  - Reduce the memory consumption of GetRegions requests #6835 @lhy1024
- TiFlash
  - Support a new DTFile format version storage.format_version = 5 to reduce the number of physical files (experimental) #7595 @hongyunyan
- Tools
  - Backup & Restore (BR)
    - When backing up data to Azure Blob Storage using BR, you can specify either an encryption scope or an encryption key for server-side encryption #45025 @Leavrth
  - TiCDC
    - Optimize the message size of the Open Protocol output to make it include only the updated column values when sending UPDATE events #9336 @3AceShowHand
    - Storage Sink now supports hexadecimal encoding for HEX formatted data, making it compatible with AWS DMS format specifications #9373 @CharlesCheung96
    - Kafka Sink supports sending only handle key data when the message is too large, reducing the size of the message #9382 @3AceShowHand

3.5 Bug fixes

- TiDB
  - Fix the issue that when the MySQL Cursor Fetch protocol is used, the memory consumption of result sets might exceed the tidb_mem_quota_query limit and causes TiDB OOM. After the fix, TiDB will automatically write result sets to the disk to release memory #43233 @YangKeao
  - Fix the TiDB panic issue caused by data race #45561 @genliqi
  - Fix the hang-up issue that occurs when queries with indexMerge are killed #45279 @xzhangxian1008
  - Fix the issue that query results in MPP mode are incorrect when tidb_enable_parallel_apply is enabled #45299 @windtalker
  - Fix the issue that resolve lock might hang when there is a sudden change in PD time #44822 @zyguan
  - Fix the issue that the GC Resolve Locks step might miss some pessimistic locks #45134 @MyonKeminta
  - Fix the issue that the query with ORDER BY returns incorrect results in dynamic pruning mode #45007 @Defined2014
  - Fix the issue that AUTO_INCREMENT can be specified on the same column with the DEFAULT column value #45136 @Defined2014
  - Fix the issue that querying the system table INFORMATION_SCHEMA.TIKV_REGION_STATUS returns incorrect results in some cases #45531 @Defined2014
  - Fix the issue of incorrect partition table pruning in some cases #42273 @jiyfhust
  - Fix the issue that global indexes are not cleared when truncating partition of a partitioned table #42435 @L-maple
  - Fix the issue that other TiDB nodes do not take over TTL tasks after failures in one TiDB node #45022 @lcwangchao
  - Fix the memory leak issue when TTL is running #45510 @lcwangchao
  - Fix the issue of inaccurate error messages when inserting data into partitioned tables #44966 @lilinghai

  - Fix the read permission issue on the INFORMATION_SCHEMA.TIFLASH_REPLICA table #7795 @Lloyd-Pottiger
  - Fix the issue that an error occurs when using a wrong partition table name #44967 @River2000i
  - Fix the issue that creating indexes gets stuck when tidb_enable_dist_task is enabled in some cases #44440 @tangenta
  - Fix the duplicate entry error that occurs when restoring a table with AUTO_ID_CACHE=1 using BR #44716 @tiancaiamao
  - Fix the issue that the time consumed for executing TRUNCATE TABLE is inconsistent with the task execution time shown in ADMIN SHOW DDL JOBS #44785 @tangenta
  - Fix the issue that upgrading TiDB gets stuck when reading metadata takes longer than one DDL lease #45176 @zimulala
  - Fix the issue that the query result of the SELECT CAST(n AS CHAR) statement is incorrect when n in the statement is a negative number #44786 @xhebox
  - Fix the issue that queries might return incorrect results when tidb_opt_agg_push_down is enabled #44795 @AilinKid
  - Fix the issue of wrong results that occurs when a query with current_date() uses plan cache #45086 @qw4990
- TiKV
  - Fix the issue that reading data during GC might cause TiKV panic in some rare cases #15109 @MyonKeminta
- PD
  - Fix the issue that restarting PD might cause the default resource group to be reinitialized #6787 @glorv
  - Fix the issue that when etcd is already started but the client has not yet connected to it, calling the client might cause PD to panic #6860 @HuSharp
  - Fix the issue that the health-check output of a Region is inconsistent with the Region information returned by querying the Region ID #6560 @JmPotato
  - Fix the issue that failed learner peers in unsafe recovery are ignored in auto-detect mode #6690 @v01dstar

  - Fix the issue that Placement Rules select TiFlash learners that do not meet the rules #6662 @rleungx
  - Fix the issue that unhealthy peers cannot be removed when rule checker selects peers #6559 @nolouch
- TiFlash
  - Fix the issue that TiFlash cannot replicate partitioned tables successfully due to deadlocks #7758 @hongyunyan
  - Fix the issue that the INFORMATION_SCHEMA.TIFLASH_REPLICA system table contains tables that users do not have privileges to access #7795 @Lloyd-Pottiger
  - Fix the issue that when there are multiple HashAgg operators within the same MPP task, the compilation of the MPP task might take an excessively long time, severely affecting query performance #7810 @SeaRise
- Tools
  - TiCDC
    - Fix the issue that changefeeds would fail due to the temporary unavailability of PD #9294 @asddongmen
    - Fix the data inconsistency issue that might occur when some TiCDC nodes are isolated from the network #9344 @CharlesCheung96
    - Fix the issue that when Kafka Sink encounters errors it might indefinitely block changefeed progress #9309 @hicqu
    - Fix the panic issue that might occur when the TiCDC node status changes #9354 @sdojjy
    - Fix the encoding error for the default ENUM values #9259 @3AceShowHand
  - TiDB Lightning
    - Fix the issue that executing checksum after TiDB Lightning completes import might get SSL errors #45462 @D3Hunter
    - Fix the issue that in Logical Import Mode, deleting tables downstream during import might cause TiDB Lightning metadata not to be updated in time #44614 @dsdashun

3.6 Contributors

We would like to thank the following contributors from the TiDB community:

- charleszheng44
- dhysum
- haiyux
- Jiang-Hua
- Jille
- jiyfhust
- krishnaduttPanchagnula
- L-maple
- pingandb
- testwill
- tisonkun
- xuyifangreeneyes
- yumchina

4 TiDB 7.2.0 Release Notes

Release date: June 29, 2023

TiDB version: 7.2.0

Quick access: Quick start | Installation packages

7.2.0 introduces the following key features and improvements:

Category	Feature	Description
Scalability and Performance	Resource groups support managing runaway queries (experimental)	You can now manage query timeout with more granularity, allowing for different behaviors based on query classifications. Queries meeting your specified threshold can be deprioritized or terminated.
	TiFlash supports the pipeline execution model (experimental)	TiFlash supports a pipeline execution model to optimize thread resource control.

Category	Feature	Description
SQL	Support a new SQL statement, IMPORT INTO, for data import (experimental)	To simplify the deployment and maintenance of TiDB Lightning, TiDB introduces a new SQL statement IMPORT INTO, which integrates physical import mode of TiDB Lightning, including remote import from Amazon S3 or Google Cloud Storage (GCS) directly into TiDB.
DB Operations and Observability	DDL supports pause and resume operations (experimental)	This new capability lets you temporarily suspend resource-intensive DDL operations, such as index creation, to conserve resources and minimize the impact on online traffic. You can seamlessly resume these operations when ready, without the need to cancel and restart. This feature enhances resource utilization, improves user experience, and streamlines schema changes.

4.1 Feature details

4.1.1 Performance

- Support pushing down the following two window functions to TiFlash #7427 @xzhangxian1008
  - FIRST_VALUE
  - LAST_VALUE
- TiFlash supports the pipeline execution model (experimental) #6518 @SeaRise

  Prior to v7.2.0, each task in the TiFlash engine must individually request thread resources during execution. TiFlash controls the number of tasks to limit thread resource usage and prevent overuse, but this issue could not be completely eliminated. To address this problem, starting from v7.2.0, TiFlash introduces a pipeline execution

model. This model centrally manages all thread resources and schedules task execution uniformly, maximizing the utilization of thread resources while avoiding resource overuse. To enable or disable the pipeline execution model, modify the tidb_enable_tiflash_pipeline_model system variable.

For more information, see documentation.

- TiFlash reduces the latency of schema replication #7630 @hongyunyan

  When the schema of a table changes, TiFlash needs to replicate the latest schema from TiKV in a timely manner. Before v7.2.0, when TiFlash accesses table data and detects a table schema change within a database, TiFlash needs to replicate the schemas of all tables in this database again, including those tables without TiFlash replicas. As a result, in a database with a large number of tables, even if you only need to read data from a single table using TiFlash, you might experience significant latency to wait for TiFlash to complete the schema replication of all tables.

  In v7.2.0, TiFlash optimizes the schema replication mechanism and supports only replicating schemas of tables with TiFlash replicas. When a schema change is detected for a table with TiFlash replicas, TiFlash only replicates the schema of that table, which reduces the latency of schema replication of TiFlash and minimizes the impact of DDL operations on TiFlash data replication. This optimization is automatically applied and does not require any manual configuration.

- Improve the performance of statistics collection #44725 @xuyifangreeneyes

  TiDB v7.2.0 optimizes the statistics collection strategy, skipping some of the duplicate information and information that is of little value to the optimizer. The overall speed of statistics collection has been improved by 30%. This improvement allows TiDB to update the statistics of the database in a more timely manner, making the generated execution plans more accurate, thus improving the overall database performance.

  By default, statistics collection skips the columns of the JSON, BLOB, MEDIUMBLOB, and LONGBLOB types. You can modify the default

behavior by setting the tidb_analyze_skip_column_types system variable. TiDB supports skipping the JSON, BLOB, and TEXT types and their subtypes.

For more information, see documentation.

- Improve the performance of checking data and index consistency #43693 @wjhuang2016

  The ADMIN CHECK [TABLE|INDEX] statement is used to check the consistency between data in a table and its corresponding indexes. In v7.2.0, TiDB optimizes the method for checking data consistency and improves the execution efficiency of ADMIN CHECK [TABLE|INDEX] greatly. In scenarios with large amounts of data, this optimization can provide a performance boost of hundreds of times.

  The optimization is enabled by default (tidb_enable_fast_table_check is ON by default) to greatly reduce the time required for data consistency checks in large-scale tables and enhance operational efficiency.

  For more information, see documentation.

4.1.2 Reliability

- Automatically manage queries that consume more resources than expected (experimental) #43691 @Connor1996 @CabinfeverB @glorv @HuSharp @nolouch

  The most common challenge to database stability is the degradation of overall database performance caused by abrupt SQL performance problems. There are many causes for SQL performance issues, such as new SQL statements that have not been fully tested, drastic changes in data volume, and abrupt changes in execution plans. These issues are difficult to completely avoid at the root. TiDB v7.2.0 provides the ability to manage queries that consume more resources than expected. This feature can quickly reduce the scope of impact when a performance issue occurs.

  To manage these queries, you can set the maximum execution time of queries for a resource group. When the execution time of a query exceeds this limit, the query is automatically deprioritized or cancelled. You can also set a period of time to immediately match

identified queries by text or execution plan. This helps prevent high concurrency of the problematic queries during the identification phase that could consume more resources than expected.

Automatic management of queries that consume more resources than expected provides you with an effective means to quickly respond to unexpected query performance problems. This feature can reduce the impact of the problem on overall database performance, thereby improving database stability.

For more information, see documentation.

- Enhance the capability of creating a binding according to a historical execution plan #39199 @qw4990

  TiDB v7.2.0 enhances the capability of creating a binding according to a historical execution plan. This feature improves the parsing and binding process for complex statements, making the bindings more stable, and supports the following new hints:

  - AGG_TO_COP()
  - LIMIT_TO_COP()
  - ORDER_INDEX
  - NO_ORDER_INDEX()

  For more information, see documentation.

- Introduce the Optimizer Fix Controls mechanism to provide fine-grained control over optimizer behaviors #43169 @time-and-fate

  To generate more reasonable execution plans, the behavior of the TiDB optimizer evolves over product iterations. However, in some particular scenarios, the changes might lead to performance regression. TiDB v7.2.0 introduces Optimizer Fix Controls to let you control some of the fine-grained behaviors of the optimizer. This enables you to roll back or control some new changes.

  Each controllable behavior is described by a GitHub issue corresponding to the fix number. All controllable behaviors are listed in Optimizer Fix Controls. You can set a target value for one or more behaviors by setting the tidb_opt_fix_control system variable to achieve behavior control.

The Optimizer Fix Controls mechanism helps you control the TiDB optimizer at a granular level. It provides a new means of fixing performance issues caused by the upgrade process and improves the stability of TiDB.

For more information, see documentation.

- Lightweight statistics initialization becomes generally available (GA) #42160 @xuyifangreeneyes

  Starting from v7.2.0, the lightweight statistics initialization feature becomes GA. Lightweight statistics initialization can significantly reduce the number of statistics that must be loaded during startup, thus improving the speed of loading statistics. This feature increases the stability of TiDB in complex runtime environments and reduces the impact on the overall service when TiDB nodes restart.

  For newly created clusters of v7.2.0 or later versions, TiDB loads lightweight statistics by default during TiDB startup and will wait for the loading to finish before providing services. For clusters upgraded from earlier versions, you can set the TiDB configuration items lite-init-stats and force-init-stats to true to enable this feature.

  For more information, see documentation.

4.1.3 SQL

- Support the CHECK constraints #41711 @fzzf678

  Starting from v7.2.0, you can use CHECK constraints to restrict the values of one or more columns in a table to meet your specified conditions. When a CHECK constraint is added to a table, TiDB checks whether the constraint is satisfied before inserting or updating data in the table. Only the data that satisfies the constraint can be written.

  This feature is disabled by default. You can set the tidb_enable_check_constraint system variable to ON to enable it.

  For more information, see documentation.

4.1.4 DB operations

- DDL jobs support pause and resume operations (experimental) #18015 @godouxm

Before TiDB v7.2.0, when a DDL job encounters a business peak during execution, you can only manually cancel the DDL job to reduce its impact on the business. In v7.2.0, TiDB introduces pause and resume operations for DDL jobs. These operations let you pause DDL jobs during a peak and resume them after the peak ends, thus avoiding impact on your application workloads.

For example, you can pause and resume multiple DDL jobs using ADMIN PAUSE DDL JOBS or ADMIN RESUME DDL JOBS:

```
ADMIN PAUSE DDL JOBS 1,2;
ADMIN RESUME DDL JOBS 1,2;
```

For more information, see documentation.

4.1.5 Data migration

- Introduce a new SQL statement IMPORT INTO to improve data import efficiency greatly (experimental) #42930 @D3Hunter

  The IMPORT INTO statement integrates the Physical Import Mode capability of TiDB Lightning. With this statement, you can quickly import data in formats such as CSV, SQL, and PARQUET into an empty table in TiDB. This import method eliminates the need for a separate deployment and management of TiDB Lightning, thereby reducing the complexity of data import and greatly improving import efficiency.

  For data files stored in Amazon S3 or GCS, when the Backend task distributed execution framework is enabled, IMPORT INTO also supports splitting a data import job into multiple sub-jobs and scheduling them to multiple TiDB nodes for parallel import, which further enhances import performance.

  For more information, see documentation.

- TiDB Lightning supports importing source files with the Latin-1 character set into TiDB #44434 @lance6716

  With this feature, you can directly import source files with the Latin-1 character set into TiDB using TiDB Lightning. Before v7.2.0, importing such files requires your additional preprocessing or conversion. Starting from v7.2.0, you only need to specify character-set = "latin1"

when configuring the TiDB Lightning import task. Then, TiDB Lightning automatically handles the character set conversion during the import process to ensure data integrity and accuracy.

For more information, see documentation.

4.2 Compatibility changes

Note:

This section provides compatibility changes you need to know when you upgrade from v7.1.0 to the current version (v7.2.0). If you are upgrading from v7.0.0 or earlier versions to the current version, you might also need to check the compatibility changes introduced in intermediate versions.

4.2.1 System variables

Variable name	Change type	Description
last_insert_id	Modified	Changes the maximum value from 9223372036854775807 to 18446744073709551615 to be consistent with that of MySQL.
tidb_enable_non_prepared_plan_cache	Modified	Changes the default value from OFF to ON after further tests, meaning that non-prepared execution plan cache is enabled.
tidb_remove_orderby_in_subquery	Modified	Changes the default value from OFF to ON after further tests, meaning that the optimizer removes the ORDER BY clause in a subquery.
tidb_analyze_skip_column_types	Newly added	Controls which types of columns are skipped for statistics collection when executing the ANALYZE command to collect statistics. The variable is only applicable for tidb_analyze_version = 2. When using the syntax of ANALYZE TABLE t COLUMNS c1, ..., cn, if the type of a specified column is included in tidb_analyze_skip_column_types, the statistics of this column will not be collected.
tidb_enable_check_constraint	Newly added	Controls whether to enable CHECK constraints. The default value is OFF, which means this feature is disabled.
tidb_enable_fast_table_check	Newly added	Controls whether to use a checksum-based approach to quickly check the consistency of data and indexes in a table. The default value is ON, which means this feature is enabled.

Variable name	Change type	Description
tidb_enable_tiflash_pipeline_model	Newly added	Controls whether to enable the new execution model of TiFlash, the pipeline model. The default value is OFF, which means the pipeline model is disabled.
tidb_expensive_txn_time_threshold	Newly added	Controls the threshold for logging expensive transactions, which is 600 seconds by default. When the duration of a transaction exceeds the threshold, and the transaction is neither committed nor rolled back, it is considered an expensive transaction and will be logged.

4.2.2 Configuration file parameters

Configuration file	Configuration parameter	Change type	Description
TiDB	lite-init-stats	Modified	Changes the default value from false to true after further tests, meaning that TiDB uses lightweight statistics initialization by default during TiDB startup to improve the initialization efficiency.
TiDB	force-init-stats	Modified	Changes the default value from false to true to align with lite-init-stats, meaning that TiDB waits for statistics initialization to finish before providing services during TiDB startup.
TiKV	rocksdb.[defaultcf|writecf|lockcf].compaction-guard-min-output-file-size	Modified	Changes the default value from "8MB" to "1MB" to reduce the data volume of compaction tasks in RocksDB.
TiKV	rocksdb.[defaultcf|writecf|lockcf].optimize-filters-for-memory	Newly added	Controls whether to generate Bloom/Ribbon filters that minimize memory internal fragmentation.
TiKV	rocksdb.[defaultcf|writecf|lockcf].periodic-compaction-seconds	Newly added	Controls the time interval for periodic compaction. SST files with updates older than this value will be selected for compaction and rewritten to the same level where these SST files originally reside.
TiKV	rocksdb.[defaultcf|writecf|lockcf].ribbon-	Newly added	Controls whether to use Ribbon filters for levels greater than or equal to this value and use non-block-based bloom filters for levels less than this value.

Configuration file	Configuration parameter	Change type	Description
	filter-above-level		
TiKV	rocksdb.[defaultcf|writecf|lockcf].ttl	Newly added	SST files with updates older than the TTL will be automatically selected for compaction.
TiDB Lightning	send-kv-pairs	Deprecated	Starting from v7.2.0, the parameter send-kv-pairs is deprecated. You can use send-kv-size to control the maximum size of one request when sending data to TiKV in physical import mode.
TiDB Lightning	character-set	Modified	Introduces a new value option latin1 for the supported character sets of data import. You can use this option to import source files with the Latin-1 character set.
TiDB Lightning	send-kv-size	Newly added	Specify the maximum size of one request when sending data to TiKV in physical import mode. When the size of key-value pairs reaches the specified threshold, TiDB Lightning will immediately send them to TiKV. This avoids the OOM problems caused by TiDB Lightning nodes accumulating too many key-value pairs in memory when importing large wide tables. By adjusting this parameter, you can find a balance between memory usage and import speed, improving the stability and efficiency of the import process.
Data Migration	strict-optimistic-shard-mode	Newly added	This configuration item is used to be compatible with the DDL shard merge behavior in TiDB Data Migration v2.0. You can enable this configuration item in optimistic mode. After this is enabled, the replication task will be interrupted when it encounters a Type 2 DDL statement. In scenarios where there are dependencies between DDL changes in multiple tables, a timely interruption can be made. You need to manually process the DDL statements of each table before resuming the replication task to ensure data consistency between the upstream and the downstream.

Configuration file	Configuration parameter	Change type	Description
TiCDC	sink.protocol	Modified	Introduces a new value option "open-protocol" when the downstream is Kafka. Specifies the protocol format used for encoding messages.
TiCDC	sink.delete-only-output-handle-key-columns	Newly added	Specifies the output of DELETE events. This parameter is valid only for "canal-json" and "open-protocol" protocols. The default value is false, which means outputting all columns. When you set it to true, only primary key columns or unique index columns are output.

4.3 Improvements

- TiDB
  - Optimize the logic of constructing index scan range so that it supports converting complex conditions into index scan range #41572 #44389 @xuyifangreeneyes
  - Add new monitoring metrics Stale Read OPS and Stale Read Traffic #43325 @you06
  - When the retry leader of stale read encounters a lock, TiDB forcibly retries with the leader after resolving the lock, which avoids unnecessary overhead #43659 @you06
  - Use estimated time to calculate stale read ts and reduce the overhead of stale read #44215 @you06
  - Add logs and system variables for long-running transactions #41471 @crazycs520
  - Support connecting to TiDB through the compressed MySQL protocol, which improves the performance of data-intensive queries under low bandwidth networks and saves bandwidth costs. This supports both zlib and zstd based compression. #22605 @dveeden
  - Recognize both utf8 and utf8bm3 as the legacy three-byte UTF-8 character set encodings, which facilitates the migration of tables with legacy UTF-8 encodings from MySQL 8.0 to TiDB #26226 @dveeden
  - Support using := for assignment in UPDATE statements #44751 @CbcWestwolf

- TiKV
    - Support configuring the retry interval of PD connections in scenarios such as connection request failures using pd.retry-interval #14964 @rleungx
    - Optimize the resource control scheduling algorithm by incorporating the global resource usage #14604 @Connor1996
    - Use gzip compression for check_leader requests to reduce traffic #14553 @you06
    - Add related metrics for check_leader requests #14658 @you06
    - Provide detailed time information during TiKV handling write commands #12362 @cfzjywxk
- PD
    - Use a separate gRPC connection for PD leader election to prevent the impact of other requests #6403 @rleungx
    - Enable the bucket splitting by default to mitigate hotspot issues in multi-Region scenarios #6433 @bufferflies
- Tools
    - Backup & Restore (BR)
        - Support access to Azure Blob Storage by shared access signature (SAS) #44199 @Leavrth
    - TiCDC
        - Optimize the structure of the directory where data files are stored when a DDL operation occurs in the scenario of replication to an object storage service #8891 @CharlesCheung96
        - Support the OAUTHBEARER authentication in the scenario of replication to Kafka #8865 @hi-rustin
        - Add the option of outputting only the handle keys for the DELETE operation in the scenario of replication to Kafka #9143 @3AceShowHand
    - TiDB Data Migration (DM)
- Support reading compressed binlogs in MySQL 8.0 as a data source for incremental replication #6381 @dveeden

- TiDB Lightning
    - Optimize the retry mechanism during import to avoid errors caused by leader switching #44478 @lance6716
    - Verify checksum through SQL after the import to improve stability of verification #41941 @GMHDBJD
    - Optimize TiDB Lightning OOM issues when importing wide tables #43853 @D3Hunter

4.4 Bug fixes

- TiDB

- Fix the issue that the query with CTE causes TiDB to hang #43749 #36896 @guo-shaoge
- Fix the issue that the min, max query result is incorrect #43805 @wshwsh12
- Fix the issue that the SHOW PROCESSLIST statement cannot display the TxnStart of the transaction of the statement with a long subquery time #40851 @crazycs520
- Fix the issue that the stale read global optimization does not take effect due to the lack of TxnScope in Coprocessor tasks #43365 @you06
- Fix the issue that follower read does not handle flashback errors before retrying, which causes query errors #43673 @you06
- Fix the issue that data and indexes are inconsistent when the ON UPDATE statement does not correctly update the primary key #44565 @zyguan
- Modify the upper limit of the UNIX_TIMESTAMP() function to 3001-01-19 03:14:07.999999 UTC to be consistent with that of MySQL 8.0.28 or later versions #43987 @YangKeao
- Fix the issue that adding an index fails in the ingest mode #44137 @tangenta
- Fix the issue that canceling a DDL task in the rollback state causes errors in related metadata #44143 @wjhuang2016
- Fix the issue that using memTracker with cursor fetch causes memory leaks #44254 @YangKeao
- Fix the issue that dropping a database causes slow GC progress #33069 @tiancaiamao

- Fix the issue that TiDB returns an error when the corresponding rows in partitioned tables cannot be found in the probe phase of index join #43686 @AilinKid @mjonss
- Fix the issue that there is no warning when using SUBPARTITION to create partitioned tables #41198 #41200 @mjonss
- Fix the issue that when a query is killed because it exceeds MAX_EXECUTION_TIME, the returned error message is inconsistent with that of MySQL #43031 @dveeden
- Fix the issue that the LEADING hint does not support querying block aliases #44645 @qw4990
- Modify the return type of the LAST_INSERT_ID() function from VARCHAR to LONGLONG to be consistent with that of MySQL #44574 @Defined2014
- Fix the issue that incorrect results might be returned when using a common table expression (CTE) in statements with non-correlated subqueries #44051 @winoros
- Fix the issue that Join Reorder might cause incorrect outer join results #44314 @AilinKid
- Fix the issue that PREPARE stmt FROM "ANALYZE TABLE xxx" might be killed by tidb_mem_quota_query #44320 @chrysan
    - TiKV

- Fix the issue that the transaction returns an incorrect value when TiKV handles stale pessimistic lock conflicts #13298 @cfzjywxk
- Fix the issue that in-memory pessimistic lock might cause flashback failures and data inconsistency #13303 @JmPotato
- Fix the issue that the fair lock might be incorrect when TiKV handles stale requests #13298 @cfzjywxk
- Fix the issue that autocommit and point get replica read might break linearizability #14715 @cfzjywxk
    - PD

- Fix the issue that redundant replicas cannot be automatically repaired in some corner cases #6573 @nolouch
    - TiFlash

- Fix the issue that queries might consume more memory than needed when the data on the Join build side is very large and contains many small string type columns #7416 @yibin87

- Tools

  - Backup & Restore (BR)

    - Fix the issue that checksum mismatch is falsely reported in some cases #44472 @Leavrth
    - Fix the issue that resolved lock timeout is falsely reported in some cases #43236 @YuJuncen
    - Fix the issue that TiDB might panic when restoring statistics information #44490 @tangenta
  - TiCDC

    - Fix the issue that Resolved TS does not advance properly in some cases #8963 @CharlesCheung96
    - Fix the issue that the UPDATE operation cannot output old values when the Avro or CSV protocol is used #9086 @3AceShowHand
    - Fix the issue of excessive downstream pressure caused by reading downstream metadata too frequently when replicating data to Kafka #8959 @hi-rustin
    - Fix the issue of too many downstream logs caused by frequently setting the downstream bidirectional replication-related variables when replicating data to TiDB or MySQL #9180 @asddongmen
    - Fix the issue that the PD node crashing causes the TiCDC node to restart #8868 @asddongmen
    - Fix the issue that TiCDC cannot create a changefeed with a downstream Kafka-on-Pulsar #8892 @hi-rustin
  - TiDB Lightning

    - Fix the TiDB Lightning panic issue when experimental.allow-expression-index is enabled and the default value is UUID #44497 @lichunzhu
    - Fix the TiDB Lightning panic issue when a task exits while dividing a data file #43195 @lance6716

4.5 Contributors

We would like to thank the following contributors from the TiDB community:

- asjdf

- blacktear23
- Cavan-xu
- darraes
- demoManito
- dhysum
- HappyUncle
- jiyfhust
- L-maple
- nyurik
- SeigeC
- tangjingyu97