TiDB Release Notes: Changes from v8.2.0 to v8.5.0

PingCAP

20241219

This document contains the release notes for v8.2.0-DMR, v8.3.0-DMR, v8.4.0-DMR, and v8.5.0-LTS. When you upgrade from v8.1.x to v8.5.0, 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 v8.5.0, see TiDB v8.5 documentation.

Table of Contents

1 TiDB 8.5.0 Release Notes

Release date: December 19, 2024

TiDB version: 8.5.0

Quick access: Quick start | Production deployment

TiDB 8.5.0 is a Long-Term Support Release (LTS).

Compared with the previous LTS 8.1.0, 8.5.0 includes new features, improvements, and bug fixes released in 8.2.0-DMR, 8.3.0-DMR, and 8.4.0-DMR. When you upgrade from 8.1.x to 8.5.0, 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 8.1.0 to 8.5.0:

Category	Feature/Enhancement	Description
Scalability and Performance	Reduce data processing latency in multiple dimensions	TiDB continuously refines data processing to improve performance, effectively meeting the low-latency SQL processing requirements in financial scenarios. Key updates include: Support parallel sorting (introduced in v8.2.0)

Category	Feature/Enhancement	Description
		Optimize batch processing strategy for KV (key-value) requests (introduced in v8.3.0) Support parallel mode for TSO requests (introduced in v8.4.0) Reduce the resource overhead of DELETE operations (introduced in v8.4.0) Improve query performance for cached tables (introduced in v8.4.0) Introduce an optimized version of Hash Join (experimental, introduced in v8.4.0)
	TiKV MVCC In-Memory Engine (IME) (introduced in v8.5.0)	The TiKV MVCC in-memory engine caches the most recent MVCC versions of data in memory, helping TiKV quickly skip older versions and retrieve the latest data. This feature can significantly improve data scan performance in scenarios where data records are frequently updated or historical versions are retained for a longer period.
	Use Active PD Followers to enhance PD's Region information query service (GA in v8.5.0)	TiDB v7.6.0 introduces an experimental feature "Active PD Follower", which allows PD followers to provide Region information query services. This feature improves the capability of the PD cluster to handle GetRegion and ScanRegions requests in clusters with a large number of TiDB nodes and

Category	Feature/Enhancement	Description
		Regions, thereby reducing the CPU pressure on PD leaders. In v8.5.0, this feature becomes generally available (GA).
	Instance-level execution plan cache (experimental, introduced in v8.4.0)	Instance-level plan cache allows all sessions within the same TiDB instance to share the plan cache. Compared with session-level plan cache, this feature reduces SQL compilation time by caching more execution plans in memory, decreasing overall SQL execution time. It improves OLTP performance and throughput while providing better control over memory usage and enhancing database stability.
	Global indexes for partitioned tables (GA in v8.4.0)	Global indexes can effectively improve the efficiency of retrieving non-partitioned columns, and remove the restriction that a unique key must contain the partition key. This feature extends the usage scenarios of TiDB partitioned tables, improves the performance of partitioned tables, and reduces resource consumption in certain query scenarios.
	Default pushdown of the Projection operator to the storage engine (introduced in v8.3.0)	Pushing the Projection operator down to the storage engine can distribute the load across storage nodes while reducing data transfer between nodes. This optimization helps to

Category	Feature/Enhancement	Description
		reduce the execution time for certain SQL queries and improves the overall database performance.
	Ignoring unnecessary columns when collecting statistics (introduced in v8.3.0)	Under the premise of ensuring that the optimizer can obtain the necessary information, TiDB speeds up statistics collection, improves the timeliness of statistics, and thus ensures that the optimal execution plan is selected, improving the performance of the cluster. Meanwhile, TiDB also reduces the system overhead and improves the resource utilization.
Reliability and availability	Improve the stability of large-scale clusters	Companies that use TiDB to run multi-tenant or SaaS applications often need to store a large number of tables. In v8.5.0, TiDB significantly enhances the stability of large-scale clusters. Schema cache control and setting the memory quota for the TiDB statistics cache are generally available (GA), reducing stability issues caused by excessive memory consumption. PD introduces Active Follower to handle the pressure brought by numerous Regions, gradually decouples the services handled by PD for independent deployment. PD improves the performance of Region heartbeat processing and supports tens of millions of Regions for a single cluster.

Category	Feature/Enhancement	Description
		You can increase concurrency and reduce the number of collected objects to improve the efficiency of statistics collection and loading, ensuring the stability of execution plans in large clusters.
	Support more triggers for runaway queries, and support switching resource groups (introduced in v8.4.0)	Runaway Queries offer an effective way to mitigate the impact of unexpected SQL performance issues on systems. TiDB v8.4.0 introduces the number of keys processed by the Coprocessor (PROCESSED_KEYS) and request units (RU) as identifying conditions, and puts identified queries into the specified resource group for more precise identification and control of runaway queries.
	Support setting the maximum limit on resource usage for background tasks of resource control (experimental, introduced in v8.4.0)	By setting a maximum percentage limit on background tasks of resource control, you can control their resource consumption based on the needs of different application systems. This keeps background task consumption at a low level and ensures the quality of online services.
	Enhance and expand TiProxy use cases	As a crucial component of the high availability of TiDB, TiProxy extends its capabilities beyond SQL traffic access and forwarding to support cluster change evaluation. Key features include:

Category	Feature/Enhancement	Description
		TiProxy supports traffic capture and replay (experimental, introduced in v8.4.0) TiProxy supports built-in virtual IP management (introduced in v8.3.0) TiProxy supports multiple load balancing policies (introduced in v8.2.0)
	The parallel HashAgg algorithm of TiDB supports disk spill (GA in v8.2.0)	HashAgg is a widely used aggregation operator in TiDB for efficiently aggregating rows with the same field values. TiDB v8.0.0 introduces parallel HashAgg as an experimental feature to further enhance processing speed. When memory resources are insufficient, parallel HashAgg spills temporary sorted data to disk, avoiding potential OOM risks caused by excessive memory usage. This improves query performance while maintaining node stability. In v8.2.0, this feature becomes generally available (GA) and is enabled by default, enabling you to safely configure the concurrency of parallel HashAgg using tidb_executor_concurrency.
SQL	Foreign key (GA in v8.5.0)	Foreign keys are constraints in a database that establish relationships between tables, ensuring data consistency and integrity. They ensure that the data referenced in a child table

Category	Feature/Enhancement	Description
		exist in the parent table, preventing the insertion of invalid data. Foreign keys also support cascading operations (such as automatic synchronization during deletion or update), simplifying business logic implementation and reducing the complexity of manually maintaining data relationships.
	Vector search (experimental, introduced in v8.4.0)	Vector search is a search method based on data semantics, which provides more relevant search results. As one of the core functions of AI and large language models (LLMs), vector search can be used in various scenarios such as Retrieval-Augmented Generation (RAG), semantic search, and recommendation systems.
DB Operations and Observability	Display TiKV and TiDB CPU times in memory tables (introduced in v8.4.0)	The CPU time is now integrated into a system table, displayed alongside other metrics for sessions or SQL, letting you observe high CPU consumption operations from multiple perspectives, and improving diagnostic efficiency. This is especially useful for diagnosing scenarios such as CPU spikes in instances or read/write hotspots in clusters.
	Support viewing aggregated TiKV CPU time by table or database (introduced in v8.4.0)	When hotspot issues are not caused by individual SQL statements, using the aggregated CPU time by table or database level in Top SQL can help you

Category	Feature/Enhancement	Description
		quickly identify the tables or applications responsible for the hotspots, significantly improving the efficiency of diagnosing hotspot and CPU consumption issues.
	Backup & Restore (BR) uses AWS SDK for Rust to access external storage (introduced in v8.5.0)	BR replaces the original Rusoto library with AWS SDK for Rust to access external storage such as Amazon S3 from TiKV. This change enhances compatibility with AWS features such as IMDSv2 and EKS Pod Identity.
Security	Client-side encryption of snapshot backup data and log backup data (GA in v8.5.0)	Before uploading backup data to your backup storage, you can encrypt the backup data to ensure its security during storage and transmission.

1.1 Feature details

1.1.1 Scalability

- Setting the memory limit for schema cache is now generally available (GA). When the number of tables reaches hundreds of thousands or even millions, this feature significantly reduces the memory usage of schema metadata #50959 @tiancaiamao @wjhuang2016 @gmhdbjd @tangenta

  In some SaaS scenarios, where the number of tables reaches hundreds of thousands or even millions, schema metadata can consume a significant amount of memory. With this feature enabled, TiDB uses the Least Recently Used (LRU) algorithm to cache and evict the corresponding schema metadata, effectively reducing memory usage.

  Starting from v8.4.0, this feature is enabled by default with a default value of 536870912 (that is, 512 MiB). You can adjust it as needed using the variable tidb_schema_cache_size.

  For more information, see documentation.

- Provide the Active PD Follower feature to enhance the scalability of PD's Region information query service (GA) #7431 @okJiang

  In a TiDB cluster with a large number of Regions, the PD leader might experience high CPU load due to the increased overhead of handling heartbeats and scheduling tasks. If the cluster has many TiDB instances, and there is a high concurrency of requests for Region information, the CPU pressure on the PD leader increases further and might cause PD services to become unavailable.

  To ensure high availability, TiDB v7.6.0 introduces Active PD Follower as an experimental feature to enhance the scalability of PD's Region information query service. In v8.5.0, this feature becomes generally available (GA). You can enable the Active PD Follower feature by setting the system variable pd_enable_follower_handle_region to ON. After this feature is enabled, TiDB evenly distributes Region information requests to all PD servers, and PD followers can also handle Region requests, thereby reducing the CPU pressure on the PD leader.

  For more information, see documentation.

1.1.2 Performance

- TiDB accelerated table creation becomes generally available (GA), significantly reducing data migration and cluster initialization time #50052 @D3Hunter @gmhdbjd

  TiDB v7.6.0 introduces accelerated table creation as an experimental feature, controlled by the system variable tidb_ddl_version. Staring from v8.0.0, this system variable is renamed to tidb_enable_fast_create_table.

  In v8.5.0, TiDB accelerated table creation becomes generally available (GA) and is enabled by default. During data migration and cluster initialization, this feature supports rapid creation of millions of tables, significantly reducing operation time.

  For more information, see documentation.

- TiKV supports the MVCC in-memory engine (IME), which accelerates queries involving scans of extensive MVCC historical versions #16141 @SpadeA-Tang @glorv @overvenus

When records are frequently updated, or TiDB is required to retain historical versions for extended periods (for example, 24 hours), the accumulation of MVCC versions can degrade scan performance. The TiKV MVCC in-memory engine improves scan performance by caching the latest MVCC versions in memory, and using a rapid GC mechanism to remove historical versions from memory.

Starting from v8.5.0, TiKV introduces MVCC in-memory engine. If the accumulation of MVCC versions in the TiKV cluster leads to degraded scan performance, you can enable the TiKV MVCC in-memory engine to improve scan performance by setting the TiKV configuration parameter in-memory-engine.enable.

For more information, see documentation.

1.1.3 Reliability

- Support limiting the maximum rate and concurrency of requests processed by PD #5739 @rleungx

  When a sudden influx of requests is sent to PD, it can lead to high workloads and potentially affect PD performance. Starting from v8.5.0, you can use pd-ctl to limit the maximum rate and concurrency of requests processed by PD, improving its stability.

  For more information, see documentation.

1.1.4 SQL

- Support foreign keys (GA) #36982 @YangKeao @crazycs520

  The foreign key feature becomes generally available (GA) in v8.5.0. Foreign key constraints help ensure data consistency and integrity. You can easily establish foreign key relationships between tables, with support for cascading updates and deletions, simplifying data management. This feature enhances support for applications with complex data relationships.

  For more information, see documentation.

- Introduce the ADMIN ALTER DDL JOBS statement to support modifying the DDL jobs online #57229 @fzzf678 @tangenta

Starting from v8.3.0, you can set the variables tidb_ddl_reorg_batch_size and tidb_ddl_reorg_worker_cnt at the session level. As a result, setting these two variables globally no longer affects all running DDL jobs. To modify the values of these variables, you need to cancel the DDL job first, adjust the variables, and then resubmit the job.

TiDB v8.5.0 introduces the ADMIN ALTER DDL JOBS statement, letting you adjust the variable values of specific DDL jobs online. This enables flexible balancing of resource consumption and performance. The changes are limited to individual jobs, making the impact more controllable. For example:

- ADMIN ALTER DDL JOBS job_id THREAD = 8;: adjusts the tidb_ddl_reorg_worker_cnt of the specified DDL job online.
- ADMIN ALTER DDL JOBS job_id BATCH_SIZE = 256;: adjusts the tidb_ddl_reorg_batch_size of the specified job online.
- ADMIN ALTER DDL JOBS job_id MAX_WRITE_SPEED = '200MiB';: adjusts the write traffic of index data to each TiKV node online.

For more information, see documentation.

1.1.5 Security

- BR supports client-side encryption of both full backup data and log backup data (GA) #28640 #56433 @joccau @Tristan1900
  - Client-side encryption of full backup data (introduced as experimental in TiDB v5.3.0) enables you to encrypt backup data on the client side using a custom fixed key.
  - Client-side encryption of log backup data (introduced as experimental in TiDB v8.4.0) enables you to encrypt log backup data on the client side using one of the following methods:
    - Encrypt using a custom fixed key
    - Encrypt using a master key stored on a local disk
    - Encrypt using a master key managed by a Key Management Service (KMS)

  Starting from v8.5.0, both encryption features become generally available (GA), offering enhanced client-side data security.

  For more information, see Encrypt the backup data and Encrypt the log backup data.

- TiKV encryption at rest supports Google Cloud Key Management Service (Google Cloud KMS) (GA) #8906 @glorv

  TiKV ensures data security by using the encryption at rest technique to encrypt stored data. The core aspect of this technique is proper key management. In v8.0.0, TiKV encryption at rest experimentally supports using Google Cloud KMS for master key management.

  Starting from v8.5.0, encryption at rest using Google Cloud KMS becomes generally available (GA). To use this feature, first create a key on Google Cloud, and then configure the [security.encryption.master-key] section in the TiKV configuration file.

  For more information, see documentation.

1.2 Compatibility changes

Note:

This section provides compatibility changes you need to know when you upgrade from v8.4.0 to the current version (v8.5.0). If you are upgrading from v8.3.0 or earlier versions to the current version, you might also need to check the compatibility changes introduced in intermediate versions.

1.2.1 Behavior changes

- In non-strict mode (sql_mode = ''), inserting NULL values into non-NULL columns now returns an error for MySQL compatibility. #55457 @joechenrh
- The ALTER TABLE ... DROP FOREIGN KEY IF EXISTS ... statement is no longer supported. #56703 @YangKeao

1.2.2 System variables

Variable name	Change type	Description
tidb_enable_fast_create_table	Modified	Changes the default value from OFF to ON after further tests, meaning that the accelerated table creation feature is enabled by default.
tidb_ddl_reorg_max_write_speed	Newly added	Limits the write bandwidth for each TiKV node and only takes effect when index creation acceleration is enabled (controlled by the tidb_ddl_enable_fast_reorg variable). For example, setting the variable

Variable name	Change type	Description
		to 200MiB limits the maximum write speed to 200 MiB/s.

1.2.3 Configuration parameters

Configuration file or component	Configuration parameter	Change type	Description
TiDB	deprecate-integer-display-length	Modified	Starting from v8.5.0, the integer display width feature is deprecated. The default value of this configuration item is changed from false to true.
TiKV	raft-client-queue-size	Modified	Changes the default value from 8192 to 16384.
PD	patrol-region-worker-count	Newly added	Controls the number of concurrent operators created by the checker when inspecting the health state of a Region.
BR	--checksum	Modified	Changes the default value from true to false, meaning that BR does not calculate the table-level checksum during full backups by default, to improve backup performance.

1.3 Operating system and platform requirement changes

Before upgrading TiDB, ensure that your operating system version meets the OS and platform requirements.

- According to CentOS Linux EOL, the upstream support for CentOS Linux 7 ends on June 30, 2024. TiDB ends the support for CentOS 7 starting from the 8.4 DMR version. It is recommended to use Rocky Linux 9.1 or a later version. Upgrading a TiDB cluster on CentOS 7 to v8.4.0 or later will cause the cluster to become unavailable.
- According to Red Hat Enterprise Linux Life Cycle, the maintenance support for Red Hat Enterprise Linux 7 ends on June 30, 2024. TiDB ends the support for Red Hat Enterprise Linux 7 starting from the 8.4 DMR version. It is recommended to use Rocky Linux 9.1 or a later

version. Upgrading a TiDB cluster on Red Hat Enterprise Linux 7 to v8.4.0 or later will cause the cluster to become unavailable.

1.4 Removed features

- The following feature has been removed:
    - In v8.4.0, TiDB Binlog is removed. Starting from v8.3.0, TiDB Binlog is fully deprecated. For incremental data replication, use TiCDC instead. For point-in-time recovery (PITR), use PITR. Before you upgrade your TiDB cluster to v8.4.0 or later versions, be sure to switch to TiCDC and PITR.
- The following features are planned for removal in future versions:
    - Starting from v8.0.0, TiDB Lightning deprecates the old version of conflict detection strategy for the physical import mode, and enables you to control the conflict detection strategy for both logical and physical import modes via the conflict.strategy parameter. The duplicate-resolution parameter for the old version of conflict detection will be removed in a future release.

1.5 Deprecated features

The following features are planned for deprecation in future versions:

- In v8.0.0, TiDB introduces the tidb_enable_auto_analyze_priority_queue system variable to control whether priority queues are enabled to optimize the ordering of tasks that automatically collect statistics. In future releases, the priority queue will be the only way to order tasks for automatically collecting statistics, so this system variable will be deprecated.
- In v7.5.0, TiDB introduces the tidb_enable_async_merge_global_stats system variable. You can use it to set TiDB to use asynchronous merging of partition statistics to avoid OOM issues. In future releases, partition statistics will be merged asynchronously, so this system variable will be deprecated.
- It is planned to redesign the automatic evolution of execution plan bindings in subsequent releases, and the related variables and behavior will change.
- In v8.0.0, TiDB introduces the tidb_enable_parallel_hashagg_spill system variable to control whether TiDB supports disk spill for the concurrent

HashAgg algorithm. In future versions, this system variable will be deprecated.

- In v5.1, TiDB introduces the tidb_partition_prune_mode system variable to control whether to enable the dynamic pruning mode for partitioned tables. Starting from v8.5.0, a warning is returned when you set this variable to static or static-only. In future versions, this system variable will be deprecated.
- The TiDB Lightning parameter conflict.max-record-rows is planned for deprecation in a future release and will be subsequently removed. This parameter will be replaced by conflict.threshold, which means that the maximum number of conflicting records is consistent with the maximum number of conflicting records that can be tolerated in a single import task.
- Starting from v6.3.0, partitioned tables use dynamic pruning mode by default. Compared with static pruning mode, dynamic pruning mode supports features such as IndexJoin and plan cache with better performance. Therefore, static pruning mode will be deprecated.

1.6 Improvements

- TiDB
    - Improve the response speed of job cancellation for the ADD INDEX acceleration feature when disabling the Distributed eXecution Framework (DXF) #56017 @lance6716
    - Improve the speed of adding indexes to small tables #54230 @tangenta
    - Add a new system variable tidb_ddl_reorg_max_write_speed to limit the maximum speed of the ingest phase when adding indexes #57156 @CbcWestwolf
    - Improve the performance of querying information_schema.tables in some cases #57295 @tangenta
    - Support dynamically adjusting more DDL job parameters #57526 @fzzf678
    - Support global indexes that contain all columns from a partition expression #56230 @Defined2014
    - Support partition pruning for list partitioned tables in range query scenarios #56673 @Defined2014

- Enable FixControl#46177 by default to fix the issue that a full table scan is incorrectly selected instead of an index range scan in some cases #46177 @terry1purcell
- Improve the internal estimation logic to better utilize statistics of multi-column and multi-value indexes, enhancing estimation accuracy for certain queries involving multi-value indexes #56915 @time-and-fate
- Improve the cost estimation for full table scans in specific scenarios, reducing the probability of incorrectly choosing a full table scan #57085 @terry1purcell
- Optimize the amount of data required for synchronous loading of statistics to improve loading performance #56812 @winoros
- Optimize the execution plan in specific cases where an OUTER JOIN involves a unique index and an ORDER BY ... LIMIT clause, improving execution efficiency #56321 @winoros

- TiKV
  - Use a separate thread to clean up replicas, ensuring stable latency for critical paths of Raft reads and writes #16001 @hbisheng
  - Improve the performance of the vector distance function by supporting SIMD #17290 @EricZequan
- PD
  - Support dynamic switching of the tso service between microservice and non-microservice modes #8477 @rleungx
  - Optimize the case format of certain fields in the pd-ctl config output #8694 @lhy1024
  - Store limit v2 becomes generally available (GA) #8865 @lhy1024
  - Support configuring Region inspection concurrency (experimental) #8866 @lhy1024
- TiFlash
  - Improve the garbage collection speed of outdated data in the background for tables with clustered indexes #9529 @JaySon-Huang
  - Improve query performance of vector search in data update scenarios #9599 @Lloyd-Pottiger

- Add monitoring metrics for CPU usage during vector index building #9032 @JaySon-Huang
- Improve the execution efficiency of logical operators #9146 @windtalker

- Tools
  - Backup & Restore (BR)
    - Reduce unnecessary log printing during backup #55902 @Leavrth
    - Optimize the error message for the encryption key --crypter.key #56388 @Tristan1900
    - Increase concurrency in BR when creating databases to improve data restore performance #56866 @Leavrth
    - Disable the table-level checksum calculation during full backups by default (--checksum=false) to improve backup performance #56373 @Tristan1900
    - Add a mechanism to independently track and reset the connection timeout for each storage node, enhancing the handling of slow nodes and preventing backup operations from hanging #57666 @3pointer
  - TiDB Data Migration (DM)
    - Add retries for DM-worker to connect to DM-master during DM cluster startup #4287 @GMHDBJD

1.7 Bug fixes

- TiDB
  - Fix the issue that TiDB does not automatically retry requests when the Region metadata returned from PD lacks Leader information, potentially causing execution errors #56757 @cfzjywxk
  - Fix the issue that TTL tasks cannot be canceled when there is a write conflict #56422 @YangKeao
  - Fix the issue that when canceling a TTL task, the corresponding SQL is not killed forcibly #56511 @lcwangchao
  - Fix the issue that existing TTL tasks are executed unexpectedly frequently in a cluster that is upgraded from v6.5 to v7.5 or later #56539 @lcwangchao

- Fix the issue that the INSERT ... ON DUPLICATE KEY statement is not compatible with mysql_insert_id #55965 @tiancaiamao
- Fix the issue that TTL might fail if TiKV is not selected as the storage engine #56402 @YangKeao
- Fix the issue that the AUTO_INCREMENT field is not correctly set after importing data using the IMPORT INTO statement #56476 @D3Hunter
- Fix the issue that TiDB does not check the index length limitation when executing ADD INDEX #56930 @fzzf678
- Fix the issue that executing RECOVER TABLE BY JOB JOB_ID; might cause TiDB to panic #55113 @crazycs520
- Fix the issue that stale read does not strictly verify the timestamp of the read operation, resulting in a small probability of affecting the consistency of the transaction when an offset exists between the TSO and the real physical time #56809 @MyonKeminta
- Fix the issue that TiDB could not resume Reorg DDL tasks from the previous progress after the DDL owner node is switched #56506 @tangenta
- Fix the issue that some metrics in the monitoring panel of Distributed eXecution Framework (DXF) are inaccurate #57172 @fzzf678 #56942 @fzzf678
- Fix the issue that REORGANIZE PARTITION fails to return error reasons in certain cases #56634 @mjonss
- Fix the issue that querying INFORMATION_SCHEMA.TABLES returns incorrect results due to case sensitivity #56987 @joechenrh
- Fix the issue of illegal memory access that might occur when a Common Table Expression (CTE) has multiple data consumers and one consumer exits without reading any data #55881 @windtalker
- Fix the issue that INDEX_HASH_JOIN might hang during an abnormal exit #54055 @wshwsh12
- Fix the issue that the TRUNCATE statement returns incorrect results when handling NULL values #53546 @tuziemon
- Fix the issue that the CAST AS CHAR function returns incorrect results due to type inference errors #56640 @zimulala

- Fix the issue of truncated strings in the output of some functions due to type inference errors #56587 @joechenrh
- Fix the issue that the ADDTIME() and SUBTIME() functions returns incorrect results when their first argument is a date type #57569 @xzhangxian1008
- Fix the issue that invalid NULL values can be inserted in non-strict mode (sql_mode = '') #56381 @joechenrh
- Fix the issue that the UPDATE statement incorrectly updates values of the ENUM type #56832 @xhebox
- Fix the issue that enabling the tidb_low_resolution_tso variable causes resource leaks during the execution of SELECT FOR UPDATE statements #55468 @tiancaiamao
- Fix the issue that the JSON_TYPE() function does not validate the parameter type, causing no errors returned when a non-JSON data type is passed #54029 @YangKeao
- Fix the issue that using JSON functions in PREPARE statements might cause execution failures #54044 @YangKeao
- Fix the issue that converting data from the BIT type to the CHAR type might cause TiKV panics #56494 @lcwangchao
- Fix the issue that using variables or parameters in the CREATE VIEW statement does not report errors #53176 @mjonss
- Fix the issue that the JSON_VALID() function returns incorrect results #56293 @YangKeao
- Fix the issue that TTL tasks are not canceled after the tidb_ttl_job_enable variable is disabled #57404 @YangKeao
- Fix the issue that using the RANGE COLUMNS partition function and the utf8mb4_0900_ai_ci collation at the same time could result in incorrect query results #57261 @Defined2014
- Fix the runtime error caused by executing a prepared statement that begins with a newline character, resulting in an array out of bounds #54283 @Defined2014
- Fix the precision issue in the UTC_TIMESTAMP() function, such as setting the precision too high #56451 @chagelo
- Fix the issue that foreign key errors are not omitted in UPDATE, INSERT, and DELETE IGNORE statements #56678 @YangKeao
- Fix the issue that when querying the information_schema.cluster_slow_query table, if the time filter is

not added, only the latest slow log file is queried #56100 @crazycs520

- Fix the issue of memory leaks in TTL tables #56934 @lcwangchao
- Fix the issue that foreign key constraints do not take effect for tables in write_only status, preventing using tables in non-public status #55813 @YangKeao
- Fix the issue that using subqueries after the NATURAL JOIN or USING clause might result in errors #53766 @dash12653
- Fix the issue that if a CTE contains the ORDER BY, LIMIT, or SELECT DISTINCT clause and is referenced by the recursive part of another CTE, it might be incorrectly inlined and result in an execution error #56603 @elsa0520
- Fix the issue that the CTE defined in VIEW is incorrectly inlined #56582 @elsa0520
- Fix the issue that Plan Replayer might report an error when importing a table structure containing foreign keys #56456 @hawkingrei
- Fix the issue that Plan Replayer might report an error when importing a table structure containing Placement Rules #54961 @hawkingrei
- Fix the issue that when using ANALYZE to collect statistics for a table, if the table contains expression indexes of virtually generated columns, the execution reports an error #57079 @hawkingrei
- Fix the issue that the DROP DATABASE statement does not correctly trigger the corresponding update in statistics #57227 @Rustin170506
- Fix the issue that when parsing a database name in CTE, it returns a wrong database name #54582 @hawkingrei
- Fix the issue that the upper bound and lower bound of the histogram are corrupted when DUMP STATS is transforming statistics into JSON #56083 @hawkingrei
- Fix the issue that EXISTS subquery results, when further involved in algebraic operations, could differ from the results in MySQL #56641 @windtalker

- Fix the issue that execution plan bindings cannot be created for the multi-table DELETE statement with aliases #56726 @hawkingrei
- Fix the issue that the optimizer does not take into account the character set and collations when simplifying complex predicates, resulting in possible execution errors #56479 @dash12653
- Fix the issue that the data in the **Stats Healthy Distribution** panel of Grafana might be incorrect #57176 @hawkingrei
- Fix the issue that vector search might return incorrect results when querying tables with clustered indexes #57627 @winoros

- TiKV
  - Fix the panic issue that occurs when read threads access outdated indexes in the MemTable of the Raft Engine #17383 @LykxSassinator
  - Fix the issue that when a large number of transactions are queuing for lock release on the same key and the key is frequently updated, excessive pressure on deadlock detection might cause TiKV OOM issues #17394 @MyonKeminta
  - Fix the issue that CPU usage for background tasks of resource control is counted twice #17603 @glorv
  - Fix the issue that TiKV OOM might occur due to the accumulation of CDC internal tasks #17696 @3AceShowHand
  - Fix the issue that large batch writes cause performance jitter when raft-entry-max-size is set too high #17701 @SpadeA-Tang
  - Fix the issue that the leader could not be quickly elected after Region split #17602 @LykxSassinator
  - Fix the issue that TiKV might panic when executing queries containing RADIANS() or DEGREES() functions #17852 @gengliqi
  - Fix the issue that write jitter might occur when all hibernated Regions are awakened #17101 @hhwyt
- PD
  - Fix the memory leak issue in hotspot cache #8698 @lhy1024
  - Fix the issue that the resource group selector does not take effect on any panel #56572 @glorv

- Fix the issue that deleted resource groups still appear in the monitoring panel #8716 @AndreMouche
- Fix unclear log descriptions during the Region syncer loading process #8717 @lhy1024
- Fix the memory leak issue in label statistics #8700 @lhy1024
- Fix the issue that configuring tidb_enable_tso_follower_proxy to 0 or OFF fails to disable the TSO Follower Proxy feature #8709 @JmPotato

- TiFlash
  - Fix the issue that the SUBSTRING() function does not support the pos and len arguments for certain integer types, causing query errors #9473 @gengliqi
  - Fix the issue that vector search performance might degrade after scaling out TiFlash write nodes in the disaggregated storage and compute architecture #9637 @kolafish
  - Fix the issue that the SUBSTRING() function returns incorrect results when the second parameter is negative #9604 @guo-shaoge
  - Fix the issue that the REPLACE() function returns an error when the first parameter is a constant #9522 @guo-shaoge
  - Fix the issue that LPAD() and RPAD() functions return incorrect results in some cases #9465 @guo-shaoge
  - Fix the issue that after creating a vector index, if the internal task for building the vector index is unexpectedly interrupted, it could result in TiFlash writing corrupted data and being unable to restart #9714 @JaySon-Huang
- Tools
  - Backup & Restore (BR)
    - Fix the OOM issue during backups when there are too many uncompleted range gaps, reducing the amount of pre-allocated memory #53529 @Leavrth
    - Fix the issue that global indexes cannot be backed up #57469 @Defined2014
    - Fix the issue that logs might print out encrypted information #57585 @kennytm

        - Fix the issue that the advancer cannot handle lock conflicts #57134 @3pointer
        - Fix potential security vulnerabilities by upgrading the k8s.io/api library version #57790 @BornChanger
        - Fix the issue that PITR tasks might return the Information schema is out of date error when there are a large number of tables in the cluster but the actual data size is small #57743 @Tristan1900
        - Fix the issue that log backup might unexpectedly enter a paused state when the advancer owner switches #58031 @3pointer
        - Fix the issue that the tiup br restore command omits checking whether the target cluster table already exists during database or table restoration, which might overwrite existing tables #58168 @RidRisR
    - TiCDC
        - Fix the issue that the Kafka messages lack Key fields when using the Debezium protocol #1799 @wk989898
        - Fix the issue that the redo module fails to properly report errors #11744 @CharlesCheung96
        - Fix the issue that TiCDC mistakenly discards DDL tasks when the schema versions of DDL tasks become non-incremental during TiDB DDL owner changes #11714 @wlwilliamx
    - TiDB Lightning
        - Fix the issue that TiDB Lightning fails to receive oversized messages sent from TiKV #56114 @fishiu
        - Fix the issue that the AUTO_INCREMENT value is set too high after importing data using the physical import mode #56814 @D3Hunter

1.8 Contributors

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

- dash12653 (First-time contributor)
- chagelo (First-time contributor)
- LindaSummer

- songzhibin97
- Hexilee

2 TiDB 8.4.0 Release Notes

Release date: November 11, 2024

TiDB version: 8.4.0

Quick access: Quick start

8.4.0 introduces the following key features and improvements:

Category	Feature/Enhancement	Description
Scalability and Performance	Instance-level execution plan cache (experimental)	Instance-level plan cache allows all sessions within the same TiDB instance to share the plan cache. Compared with session-level plan cache, this feature reduces SQL compilation time by caching more execution plans in memory, decreasing overall SQL execution time. It improves OLTP performance and throughput while providing better control over memory usage and enhancing database stability.
	Global indexes for partitioned tables (GA)	Global indexes can effectively improve the efficiency of retrieving non-partitioned columns, and remove the restriction that a unique key must contain the partition key. This feature extends the usage scenarios of TiDB partitioned tables, and avoids some of the application modification work required for data migration.
	Parallel mode for TSO requests	In high-concurrency scenarios, you can use this feature to reduce the wait time for retrieving TSO and improve the cluster throughput.
	Improve query performance for cached tables	Improve query performance for index scanning on cached tables, with improvements of up to 5.4 times in some scenarios. For high-speed queries on small tables, using cached tables can significantly enhance overall performance.
Reliability and Availability	Support more triggers for runaway queries, and support switching resource groups	Runaway Queries offer an effective way to mitigate the impact of unexpected SQL performance issues on systems. TiDB v8.4.0 introduces the number of keys processed by

Category	Feature/Enhancement	Description
		the Coprocessor (PROCESSED_KEYS) and request units (RU) as identifying conditions, and puts identified queries into the specified resource group for more precise identification and control of runaway queries.
	Support setting the maximum limit on resource usage for background tasks of resource control	By setting a maximum percentage limit on background tasks of resource control, you can control their resource consumption based on the needs of different application systems. This keeps background task consumption at a low level and ensures the quality of online services.
	TiProxy supports traffic capture and replay (experimental)	Use TiProxy to capture real workloads from TiDB production clusters before major operations such as cluster upgrades, migrations, or deployment changes. Replay these workloads on target test clusters to validate performance and ensure successful changes.
	Concurrent automatic statistics collection	You can set the concurrency within a single automatic statistics collection task using the system variable tidb_auto_analyze_concurrency. TiDB automatically determines the concurrency of scanning tasks based on node scale and hardware specifications. This improves statistics collection efficiency by fully utilizing system resources, reduces manual tuning, and ensures stable cluster performance.
SQL	Vector search (experimental)	Vector search is a search method based on data semantics, which provides more relevant search results. As one of the core functions of AI and large language models (LLMs), vector search can be used in various scenarios such as Retrieval-Augmented Generation (RAG), semantic search, and recommendation systems.
DB Operations and Observability	Display TiKV and TiDB CPU times in memory tables	The CPU time is now integrated into a system table, displayed alongside other metrics for sessions or SQL, letting you observe high CPU consumption operations from multiple perspectives, and improving diagnostic efficiency. This is especially useful for

Category	Feature/Enhancement	Description
		diagnosing scenarios such as CPU spikes in instances or read/write hotspots in clusters.
	Support viewing aggregated TiKV CPU time by table or database	When hotspot issues are not caused by individual SQL statements, using the aggregated CPU time by table or database level in Top SQL can help you quickly identify the tables or applications responsible for the hotspots, significantly improving the efficiency of diagnosing hotspot and CPU consumption issues.
	Support backing up TiKV instances with IMDSv2 service enabled	AWS EC2 now uses IMDSv2 as the default metadata service. TiDB supports backing up data from TiKV instances that have IMDSv2 enabled, helping you run TiDB clusters more effectively in public cloud services.
Security	Client-side encryption of log backup data (experimental)	Before uploading log backup data to your backup storage, you can encrypt the backup data to ensure its security during storage and transmission.

2.1 Feature details

2.1.1 Performance

- Introduce parallel batching modes for TSO requests, reducing TSO retrieval latency #54960 #8432 @MyonKeminta

  Before v8.4.0, when requesting TSO from PD, TiDB collects multiple TSO requests during a specific period and processes them in batches serially to decrease the number of Remote Procedure Call (RPC) requests and reduce PD workload. In latency-sensitive scenarios, however, the performance of this serial batching mode is not ideal.

  In v8.4.0, TiDB introduces parallel batching modes for TSO requests with different concurrency capabilities. Parallel modes reduce TSO retrieval latency but might increase the PD workload. To set a parallel RPC mode for retrieving TSO, configure the tidb_tso_client_rpc_mode system variable.

  For more information, see documentation.

- Optimize the execution efficiency of the hash join operator for TiDB (experimental) #55153 #53127 @windtalker @xzhangxian1008 @XuHuaiyu @wshwsh12

In v8.4.0, TiDB introduces an optimized version of the hash join operator to improve its execution efficiency. Currently, the optimized version of the hash join applies only to inner join and outer join operations and is disabled by default. To enable this optimized version, configure the tidb_hash_join_version system variable to optimized.

For more information, see documentation.

- Support pushing down the following date functions to TiKV #56297 #17529 @gengliqi
    - DATE_ADD()
    - DATE_SUB()
    - ADDDATE()
    - SUBDATE()

  For more information, see documentation.

- Support instance-level execution plan cache (experimental) #54057 @qw4990

  Instance-level execution plan cache allows all sessions within the same TiDB instance to share the execution plan cache. This feature significantly reduces TiDB query response time, increases cluster throughput, decreases the possibility of execution plan mutations, and maintains stable cluster performance. Compared with session-level execution plan cache, instance-level execution plan cache offers the following advantages:

    - Eliminates redundancy, caching more execution plans with the same memory consumption.
    - Allocates a fixed-size memory on the instance, limiting memory usage more effectively.

  In v8.4.0, instance-level execution plan cache only supports caching query execution plans and is disabled by default. You can enable this feature using tidb_enable_instance_plan_cache and set its maximum memory usage using tidb_instance_plan_cache_max_size. Before enabling this feature, disable Prepared execution plan cache and Non-prepared execution plan cache.

  For more information, see documentation.

- TiDB Lightning's logical import mode supports prepared statements and client statement cache #54850 @dbsid

  By enabling the logical-import-prep-stmt configuration item, the SQL statements executed in TiDB Lightning's logical import mode will use prepared statements and client statement cache. This reduces the cost of TiDB SQL parsing and compilation, improves SQL execution efficiency, and increases the likelihood of hitting the execution plan cache, thereby speeding up logical import.

  For more information, see documentation.

- Partitioned tables support global indexes (GA) #45133 @mjonss @Defined2014 @jiyfhust @L-maple

  In early TiDB versions, the partitioned table has some limitations because it does not support global indexes. For example, the unique key must use every column in the table's partition expression. If the query condition does not use the partition key, the query will scan all partitions, resulting in poor performance. Starting from v7.6.0, the system variable tidb_enable_global_index is introduced to enable the global index feature. But this feature was under development at that time and it is not recommended to enable it.

  Starting from v8.3.0, the global index feature is released as an experimental feature. You can explicitly create a global index for a partitioned table with the GLOBAL keyword. This removes the restriction that a unique key in a partitioned table must include all columns used in the partition expression, allowing for more flexible application requirements. Additionally, global indexes also improve the performance of queries based on non-partitioned columns.

  In v8.4.0, this feature becomes generally available (GA). You can use the keyword GLOBAL to create a global index, instead of setting the system variable tidb_enable_global_index to enable the global index feature. Starting from v8.4.0, this system variable is deprecated and is always ON.

  For more information, see documentation.

- Improve query performance for cached tables in some scenarios #43249 @tiancaiamao

In v8.4.0, TiDB improves the query performance of cached tables by up to 5.4 times when executing SELECT ... LIMIT 1 with IndexLookup. In addition, TiDB improves the performance of IndexLookupReader in full table scan and primary key query scenarios.

2.1.2 Reliability

- Runaway queries support the number of processed keys and request units as thresholds #54434 @HuSharp

  Starting from v8.4.0, TiDB can identify runaway queries based on the number of processed keys (PROCESSED_KEYS) and request units (RU). Compared with execution time (EXEC_ELAPSED), these new thresholds more accurately define the resource consumption of queries, avoiding identification bias when overall performance decreases.

  You can set multiple conditions simultaneously, and a query is identified as a runaway query if any condition is met.

  You can observe the corresponding fields (RESOURCE_GROUP, MAX_REQUEST_UNIT_WRITE, MAX_REQUEST_UNIT_READ, MAX_PROCESSED_KEYS) in the Statement Summary Tables to determine the condition values based on historical execution.

  For more information, see documentation.

- Support switching resource groups for runaway queries #54434 @JmPotato

  Starting from TiDB v8.4.0, you can switch the resource group of runaway queries to a specific one. If the COOLDOWN mechanism fails to lower resource consumption, you can create a resource group, limit its resource size, and set the SWITCH_GROUP parameter to move identified runaway queries to this group. Meanwhile, subsequent queries within the same session will continue to execute in the original resource group. By switching resource groups, you can manage resource usage more precisely, and control the resource consumption more strictly.

  For more information, see documentation.

- Support setting the cluster-level Region scattering strategy using the tidb_scatter_region system variable #55184 @D3Hunter

Before v8.4.0, the tidb_scatter_region system variable can only be enabled or disabled. When it is enabled, TiDB applies a table-level scattering strategy during batch table creation. However, when creating hundreds of thousands of tables in a batch, this strategy results in a concentration of Regions in a few TiKV nodes, causing OOM (Out of Memory) issues in those nodes.

Starting from v8.4.0, tidb_scatter_region is changed to the string type. It now supports a cluster-level scattering strategy, which can help avoid TiKV OOM issues in the preceding scenario.

For more information, see documentation.

- Support setting the maximum limit on resource usage for background tasks of resource control #56019 @glorv

  TiDB resource control can identify and lower the priority of background tasks. In certain scenarios, you might want to limit the resource consumption of background tasks, even when resources are available. Starting from v8.4.0, you can use the UTILIZATION_LIMIT parameter to set the maximum percentage of resources that background tasks can consume. Each node will keep the resource usage of all background tasks below this percentage. This feature enables precise control over resource consumption for background tasks, further enhancing cluster stability.

  For more information, see documentation.

- Optimize the resource allocation strategy of resource groups #50831 @nolouch

  TiDB improves the resource allocation strategy in v8.4.0 to better meet user expectations for resource management.

  - Controlling the resource allocation of large queries at runtime to avoid exceeding the resource group limit, combined with runaway queries COOLDOWN. This can help identify and reduce the concurrency of large queries, and reduce instantaneous resource consumption.
  - Adjusting the default priority scheduling strategy. When tasks of different priorities run simultaneously, high-priority tasks receive more resources.

2.1.3 Availability

- TiProxy supports traffic replay (experimental) #642 @djshow832

  Starting from TiProxy v1.3.0, you can use tiproxyctl to connect to the TiProxy instance, capture access traffic in a TiDB production cluster, and replay it in a test cluster at a specified rate. This feature enables you to reproduce actual workloads from the production cluster in a test environment, verifying SQL statement execution results and performance.

  Traffic replay is useful in the following scenarios:

  - Verify TiDB version upgrades
  - Assess change impact
  - Validate performance before scaling TiDB
  - Test performance limits

  For more information, see documentation.

2.1.4 SQL

- Support vector search (experimental) #54245 #17290 #9032 @breezewish @Lloyd-Pottiger @EricZequan @zimulala @JaySon-Huang @winoros @wk989898

  Vector search is a search method based on data semantics, which provides more relevant search results. As one of the core functions of AI and large language models (LLMs), vector search can be used in various scenarios such as Retrieval-Augmented Generation (RAG), semantic search, and recommendation systems.

  Starting from v8.4.0, TiDB supports vector data types and vector search indexes, offering powerful vector search capabilities. TiDB vector data types support up to 16,383 dimensions and support various distance functions, including L2 distance (Euclidean distance), cosine distance, negative inner product, and L1 distance (Manhattan distance).

  To start vector search, you only need to create a table with vector data types, insert vector data, and then perform a query of vector data. You can also perform mixed queries of vector data and traditional relational data.

To enhance the performance of vector search, you can create and use vector search indexes. Note that TiDB vector search indexes rely on TiFlash. Before using vector search indexes, make sure that TiFlash nodes are deployed in your TiDB cluster.

For more information, see documentation.

2.1.5 DB operations

- BR supports client-side encryption of log backup data (experimental) #55834 @Tristan1900

  In earlier TiDB versions, only snapshot backup data can be encrypted on the client side. Starting from v8.4.0, log backup data can also be encrypted on the client side. Before uploading log backup data to your backup storage, you can encrypt the backup data to ensure its security via one of the following methods:

  - Encrypt using a custom fixed key
  - Encrypt using a master key stored on a local disk
  - Encrypt using a master key managed by a Key Management Service (KMS)

  For more information, see documentation.

- BR requires fewer privileges when restoring backup data in a cloud storage system #55870 @Leavrth

  Before v8.4.0, BR writes checkpoint information about the restore progress to the backup storage system during restore. These checkpoints enable quick resumption of interrupted restores. Starting from v8.4.0, BR writes restore checkpoint information to the target TiDB cluster instead. This means that BR only requires read access to the backup directories during restore.

  For more information, see documentation.

2.1.6 Observability

- Display the CPU time consumed by TiDB and TiKV in the system table #55542 @yibin87

  The Top SQL page of TiDB Dashboard displays SQL statements with high CPU consumption. Starting from v8.4.0, TiDB adds CPU time consumption information to the system table, presented alongside

other metrics for sessions or SQL, making it easier to observe high CPU consumption operations from multiple perspectives. This information can help you quickly identify the causes of issues in scenarios like instance CPU spikes or read/write hotspots in clusters.

    - The statement summary tables add AVG_TIDB_CPU_TIME and AVG_TIKV_CPU_TIME, showing the average CPU time consumed by individual SQL statements historically.
    - The INFORMATION_SCHEMA.PROCESSLIST table adds TIDB_CPU and TIKV_CPU, showing the cumulative CPU consumption of the SQL statements currently being executed in a session.
    - The slow query log adds the Tidb_cpu_time and Tikv_cpu_time fields, showing the CPU time consumed by captured SQL statements.

  By default, the CPU time consumed by TiKV is displayed. Collecting the CPU time consumed by TiDB brings additional overhead (about 8%), so the CPU time consumed by TiDB only shows the actual value when Top SQL is enabled; otherwise, it always shows as 0.

  For more information, see INFORMATION_SCHEMA.PROCESSLIST and INFORMATION_SCHEMA.SLOW_QUERY.

- Top SQL supports viewing aggregated CPU time results by table or database #55540 @nolouch

  Before v8.4.0, Top SQL aggregates CPU time by SQL. If CPU time is not consumed by a few SQL statements, aggregation by SQL cannot effectively identify issues. Starting from v8.4.0, you can choose to aggregate CPU time **By TABLE** or **By DB**. In scenarios with multiple systems, the new aggregation method can more effectively identify load changes from a specific system, improving diagnostic efficiency.

  For more information, see documentation.

2.1.7 Security

- BR supports AWS IMDSv2 #16443 @pingyu

  When deploying TiDB on Amazon EC2, BR supports AWS Instance Metadata Service Version 2 (IMDSv2). You can configure your EC2 instance to allow BR to use the IAM role associated with the instance for appropriate permissions to access Amazon S3.

For more information, see documentation.

2.1.8 Data migration

- TiCDC Claim-Check supports sending only the value field of Kafka messages to external storage #11396 @3AceShowHand

  Before v8.4.0, when the Claim-Check feature is enabled (by setting large-message-handle-option to claim-check), TiCDC encodes and stores both the key and value fields in the external storage system when handling large messages.

  Starting from v8.4.0, TiCDC supports sending only the value field of Kafka messages to external storage. This feature is only applicable to non-Open Protocol protocols. You can control this feature by setting the claim-check-raw-value parameter.

  For more information, see documentation.

- TiCDC introduces Checksum V2 to verify old values in Update or Delete events #10969 @3AceShowHand

  Starting from v8.4.0, TiDB and TiCDC introduce the Checksum V2 algorithm to address issues of Checksum V1 in verifying old values in Update or Delete events after ADD COLUMN or DROP COLUMN operations. For clusters created in v8.4.0 or later, or clusters upgraded to v8.4.0, TiDB uses Checksum V2 by default when single-row data checksum verification is enabled. TiCDC supports handling both Checksum V1 and V2. This change only affects TiDB and TiCDC internal implementation and does not affect checksum calculation methods for downstream Kafka consumers.

  For more information, see documentation.

2.2 Compatibility changes

Note:

This section provides compatibility changes you need to know when you upgrade from v8.3.0 to the current version (v8.4.0). If you are upgrading from v8.2.0 or earlier versions to the current version, you might also need to check the compatibility changes introduced in intermediate versions.

2.2.1 System variables

Variable name	Change type	Description
log_bin	Deleted	In v8.4.0, TiDB Binlog is removed. This variable indicates whether TiDB Binlog is used, and is deleted starting from v8.4.0.
sql_log_bin	Deleted	In v8.4.0, TiDB Binlog is removed. This variable indicates whether to write changes to TiDB Binlog or not, and is deleted starting from v8.4.0.
tidb_enable_global_index	Deprecated	In v8.4.0, this variable is deprecated. Its value will be fixed to the default value ON, that is, global index is enabled by default. You only need to add the keyword GLOBAL to the corresponding column when executing CREATE TABLE or ALTER TABLE to create a global index.
tidb_enable_list_partition	Deprecated	In v8.4.0, this variable is deprecated. Its value will be fixed to the default value ON, that is, list partitioning is enabled by default.
tidb_enable_table_partition	Deprecated	In v8.4.0, this variable is deprecated. Its value will be fixed to the default value ON, that is, table partitioning is enabled by default.
tidb_analyze_partition_concurrency	Modified	Changes the value range from [1, 18446744073709551615] to [1, 128].
tidb_enable_inl_join_inner_multi_pattern	Modified	Changes the default value from OFF to ON. Starting from v8.4.0, Index Join is supported by default when the inner table has Selection, Aggregation, or Projection operators on it.
tidb_opt_prefer_range_scan	Modified	Changes the default value from OFF to ON. For tables with no statistics (pseudo-statistics) or empty tables (zero statistics), the optimizer prefers interval scans over full table scans.
tidb_scatter_region	Modified	Before v8.4.0, its type is boolean, it only supports ON and OFF, and the Region of the newly created table only supports table level scattering after it is enabled. Starting from v8.4.0, the SESSION scope is added, the type is changed from boolean to

Variable name	Change type	Description
		enumeration, the default value is changed from OFF to null, and the optional values TABLE and GLOBAL are added. In addition, it now supports cluster-level scattering policy to avoid the TiKV OOM issues caused by uneven distribution of regions during fast table creation in batches.
tidb_schema_cache_size	Modified	Changes the default value from 0 to 536870912 (512 MiB), indicating that this feature is enabled by default. The minimum value allowed is set to 67108864 (64 MiB).
tidb_auto_analyze_concurrency	Newly added	Sets the concurrency within a single automatic statistics collection task. Before v8.4.0, this concurrency is fixed at 1. To speed up statistics collection tasks, you can increase this concurrency based on your cluster's available resources.
tidb_enable_instance_plan_cache	Newly added	Controls whether to enable the Instance Plan Cache feature.
tidb_enable_stats_owner	Newly added	Controls whether the corresponding TiDB instance can run automatic statistics update tasks.
tidb_hash_join_version	Newly added	Controls whether TiDB uses an optimized version of the Hash Join operator. The default value of legacy means that the optimized version is not used. If you set it to optimized, TiDB uses the optimized version of the Hash Join operator when executing it to improve Hash Join performance.
tidb_instance_plan_cache_max_size	Newly added	Sets the maximum memory usage for Instance Plan Cache.
tidb_instance_plan_cache_reserved_percentage	Newly added	Controls the percentage of idle memory reserved for Instance Plan Cache after memory eviction.
tidb_pre_split_regions	Newly added	Before v8.4.0, setting the default number of row split slices for newly created tables required declaring PRE_SPLIT_REGIONS in each CREATE TABLE SQL statement, which is complicated once a large number of tables need to be similarly configured. This variable is introduced to solve such problems. You can set this system variable

Variable name	Change type	Description
		at the GLOBAL or SESSION level to improve usability.
tidb_shard_row_id_bits	Newly added	Before v8.4.0, setting the default number of slices for row IDs for newly created tables required declaring SHARD_ROW_ID_BITS in each CREATE TABLE or ALTER TABLE SQL statement, which is complicated once a large number of tables need to be similarly configured. This variable is introduced to solve such problems. You can set this system variable at the GLOBAL or SESSION level to improve usability.
tidb_tso_client_rpc_mode	Newly added	Switches the mode in which TiDB sends TSO RPC requests to PD. The mode determines whether TSO RPC requests can be processed in parallel and affects the time spent on batch-waiting for each TS retrieval operation, thereby helping reduce the wait time for retrieving TS during the execution of queries in certain scenarios.

2.2.2 Configuration parameters

Configuration file or component	Configuration parameter	Change type	Description
TiDB	grpc-keepalive-time	Modified	Adds the minimum value of 1.
TiDB	grpc-keepalive-timeout	Modified	Before v8.4.0, the data type of this parameter is INT, and the minimum value is 1. Starting from v8.4.0, the data type is changed to FLOAT64, and the minimum value becomes 0.05. In scenarios where network jitter occurs frequently, you can reduce the impact of network jitter on performance by setting a smaller value to shorten the retry interval.
TiDB	tidb_enable_stats_owner	Newly added	Controls whether the corresponding TiDB

Configuration file or component	Configuration parameter	Change type	Description
			instance can run automatic statistics update tasks.
TiKV	region-split-keys	Modified	Changes the default value from "960000" to "2560000".
TiKV	region-split-size	Modified	Changes the default value from "96MiB" to "256MiB".
TiKV	sst-max-size	Modified	Changes the default value from "144MiB" to "384MiB".
TiKV	pessimistic-txn.in-memory-instance-size-limit	Newly added	Controls the memory usage limit for in-memory pessimistic locks in a TiKV instance. When this limit is exceeded, TiKV writes pessimistic locks persistently.
TiKV	pessimistic-txn.in-memory-peer-size-limit	Newly added	Controls the memory usage limit for in-memory pessimistic locks in a Region. When this limit is exceeded, TiKV writes pessimistic locks persistently.
TiKV	raft-engine.spill-dir	Newly added	Controls the secondary directory where TiKV instances store Raft log files for supporting multi-disk storage of Raft log files.
TiKV	resource-control.priority-ctl-strategy	Newly added	Controls the management policies for low priority tasks. TiKV ensures that higher priority tasks are executed first by adding flow control to low priority tasks.
PD	cert-allowed-cn	Modified	Starting from v8.4.0, configuring multiple Common Names is supported. Before v8.4.0,

Configuration file or component	Configuration parameter	Change type	Description
			only one Common Name can be set.
PD	max-merge-region-keys	Modified	Changes the default value from 200000 to 540000.
PD	max-merge-region-size	Modified	Changes the default value from 20 to 54.
TiFlash	storage.format_version	Modified	Changes the default TiFlash storage format version from 5 to 7 to support vector index creation and storage. Due to this format change, TiFlash clusters upgraded to v8.4.0 or a later version do not support in-place downgrading to earlier versions.
TiDB Binlog	--enable-binlog	Deleted	In v8.4.0, TiDB Binlog is removed. This parameter controls whether to enable TiDB binlog generation or not, and is deleted starting from v8.4.0.
TiCDC	claim-check-raw-value	Newly added	Controls whether TiCDC sends only the value field of Kafka messages to external storage. This feature is only applicable to non-Open Protocol scenarios.
TiDB Lightning	logical-import-prep-stmt	Newly added	In Logical Import Mode, this parameter controls whether to use prepared statements and statement cache to improve performance. The default value is false.
BR	--log.crypter.key	Newly added	Specifies the encryption key in hexadecimal string format for log backup data. It is a 128-bit (16 bytes) key for the algorithm aes128-

Configuration file or component	Configuration parameter	Change type	Description
			ctr, a 24-byte key for the algorithm aes192-ctr, and a 32-byte key for the algorithm aes256-ctr.
BR	--log.crypter.key-file	Newly added	Specifies the key file for log backup data. You can directly pass in the file path where the key is stored as a parameter without passing in the crypter.key.
BR	--log.crypter.method	Newly added	Specifies the encryption algorithm for log backup data, which can be aes128-ctr, aes192-ctr, or aes256-ctr. The default value is plaintext, indicating that data is not encrypted.
BR	--master-key	Newly added	Specifies the master key for log backup data. It can be a master key stored on a local disk or a master key managed by a cloud Key Management Service (KMS).
BR	--master-key-crypter-method	Newly added	Specifies the encryption algorithm based on the master key for log backup data, which can be aes128-ctr, aes192-ctr, or aes256-ctr. The default value is plaintext, indicating that data is not encrypted.

2.3 Offline package changes

Starting from v8.4.0, the following contents are removed from the TiDB-community-toolkit binary package:

- pump-{version}-linux-{arch}.tar.gz
- drainer-{version}-linux-{arch}.tar.gz
- binlogctl
- arbiter

2.4 Operating system and platform requirement changes

Before upgrading TiDB, ensure that your operating system version meets the OS and platform requirements.

- According to CentOS Linux EOL, the upstream support for CentOS Linux 7 ends on June 30, 2024. TiDB ends the support for CentOS 7 starting from the 8.4 DMR version. It is recommended to use Rocky Linux 9.1 or a later version. Upgrading a TiDB cluster on CentOS 7 to v8.4.0 or later will cause the cluster to become unavailable.
- According to Red Hat Enterprise Linux Life Cycle, the maintenance support for Red Hat Enterprise Linux 7 ends on June 30, 2024. TiDB ends the support for Red Hat Enterprise Linux 7 starting from the 8.4 DMR version. It is recommended to use Rocky Linux 9.1 or a later version. Upgrading a TiDB cluster on Red Hat Enterprise Linux 7 to v8.4.0 or later will cause the cluster to become unavailable.

2.5 Removed features

- The following features are removed starting from v8.4.0:
    - In v8.4.0, TiDB Binlog is removed. Starting from v8.3.0, TiDB Binlog is fully deprecated. For incremental data replication, use TiCDC instead. For point-in-time recovery (PITR), use PITR. Before you upgrade your TiDB cluster to v8.4.0 or later versions, be sure to switch to TiCDC and PITR.
- The following features are planned for removal in future versions:
    - Starting from v8.0.0, TiDB Lightning deprecates the old version of conflict detection strategy for the physical import mode, and enables you to control the conflict detection strategy for both logical and physical import modes via the conflict.strategy parameter. The duplicate-resolution parameter for the old version of conflict detection will be removed in a future release.

2.6 Deprecated features

The following features are planned for deprecation in future versions:

- TiDB introduces the system variable tidb_enable_auto_analyze_priority_queue, which controls whether priority queues are enabled to optimize the ordering of tasks that automatically collect statistics. In future releases, the priority queue

will be the only way to order tasks for automatically collecting statistics, so this system variable will be deprecated.

- TiDB introduces the system variable tidb_enable_async_merge_global_stats in v7.5.0. You can use it to set TiDB to use asynchronous merging of partition statistics to avoid OOM issues. In future releases, partition statistics will be merged asynchronously, so this system variable will be deprecated.
- It is planned to redesign the automatic evolution of execution plan bindings in subsequent releases, and the related variables and behavior will change.
- In v8.0.0, TiDB introduces the tidb_enable_parallel_hashagg_spill system variable to control whether TiDB supports disk spill for the concurrent HashAgg algorithm. In future versions, the tidb_enable_parallel_hashagg_spill system variable will be deprecated.
- The TiDB Lightning parameter conflict.max-record-rows is planned for deprecation in a future release and will be subsequently removed. This parameter will be replaced by conflict.threshold, which means that the maximum number of conflicting records is consistent with the maximum number of conflicting records that can be tolerated in a single import task.
- Starting from v6.3.0, partitioned tables use dynamic pruning mode by default. Compared with static pruning mode, dynamic pruning mode supports features such as IndexJoin and plan cache with better performance. Therefore, static pruning mode will be deprecated.

2.7 Improvements

- TiDB
    - Optimize the efficiency of constructing BatchCop tasks when scanning a large amount of data #55915 #55413 @wshwsh12
    - Optimize the transaction's buffer to reduce write latency in transactions and TiDB CPU usage #55287 @you06
    - Optimize the execution performance of DML statements when the system variable tidb_dml_type is set to "bulk" #50215 @ekexium
    - Support using Optimizer Fix Control 47400 to control whether the optimizer limits the minimum value estimated for estRows to 1, which is consistent with databases such as Oracle and DB2 #47400 @terry1purcell

- Add write control to the mysql.tidb_runaway_queries log table to reduce overhead caused by a large number of concurrent writes #54434 @HuSharp
- Support Index Join by default when the inner table has Selection, Projection, or Aggregation operators on it #47233 @winoros
- Reduce the number of column details fetched from TiKV for DELETE operations in certain scenarios, lowering the resource overhead of these operations #38911 @winoros
- Support setting the concurrency within a single automatic statistics collection task using the system variable tidb_auto_analyze_concurrency #53460 @hawkingrei
- Optimize the logic of an internal function to improve performance when querying tables with numerous columns #52112 @Rustin170506
- Simplify filter conditions like a = 1 AND (a > 1 OR (a = 1 AND b = 2)) to a = 1 AND b = 2 #56005 @ghazalfamilyusa
- Increase the cost of table scans in the cost model for scenarios with a high risk of suboptimal execution plans, making the optimizer prefer indexes #56012 @terry1purcell
- TiDB supports the two-argument variant MID(str, pos) #52420 @dveeden
- Support splitting TTL tasks for tables with non-binary primary keys #55660 @lcwangchao
- Optimize performance of system metadata-related statements #50305 @ywqzzy @tangenta @joechenrh @CbcWestwolf
- Implement a new priority queue for auto-analyze operations to improve analyze performance and reduce the cost of rebuilding the queue #55906 @Rustin170506
- Introduce a DDL notifier to allow the statistics module to subscribe to DDL events #55722 @fzzf678 @lance6716 @Rustin170506
- Force new TiDB nodes to take over DDL ownership during TiDB upgrades to avoid compatibility issues caused by old TiDB nodes taking ownership #51285 @wjhuang2016
- Support cluster-level Scatter Region #8424 @River2000i

- TiKV

- Increase the default value of Region from 96 MiB to 256 MiB to avoid the extra overhead caused by too many Regions #17309 @LykxSassinator
- Support setting memory usage limits for in-memory pessimistic locks in a Region or TiKV instance. When hot write scenarios cause a large number of pessimistic locks, you can increase the memory limits via configuration. This helps avoid CPU and I/O overhead caused by pessimistic locks being written to disk. #17542 @cfzjywxk
- Introduce a new spill-dir configuration item in Raft Engine, supporting multi-disk storage for Raft logs; when the disk where the home directory (dir) is located runs out of space, the Raft Engine automatically writes new logs to spill-dir, ensuring continuous operation of the system #17356 @LykxSassinator
- Optimize the compaction trigger mechanism of RocksDB to accelerate disk space reclamation when handling a large number of DELETE versions #17269 @AndreMouche
- Support dynamically modifying flow-control configurations for write operations #17395 @glorv
- Improve the speed of Region Merge in scenarios with empty tables and small Regions #17376 @LykxSassinator
- Prevent Pipelined DML from blocking resolved-ts for long periods #17459 @ekexium

- PD
    - Support graceful offline of TiKV nodes during data import by TiDB Lightning #7853 @okJiang
    - Rename scatter-range to scatter-range-scheduler in pd-ctl commands #8379 @okJiang
    - Add conflict detection for grant-hot-leader-scheduler #4903 @lhy1024
- TiFlash
    - Optimize the execution efficiency of LENGTH() and ASCII() functions #9344 @xzhangxian1008
    - Reduce the number of threads that TiFlash needs to create when processing disaggregated storage and compute requests, helping avoid crashes of TiFlash compute nodes when processing a large number of such requests #9334 @JinheLin

- Enhance the task waiting mechanism in the pipeline execution model #8869 @SeaRise
- Improve the cancel mechanism of the JOIN operator, so that the JOIN operator can respond to cancel requests in a timely manner #9430 @windtalker

- Tools
  - Backup & Restore (BR)
    - Disable splitting Regions by table to improve restore speed when restoring data to a cluster where the split-table and split-region-on-table configuration items are false (default value) #53532 @Leavrth
    - Disable full data restoration to a non-empty cluster using the RESTORE SQL statement by default #55087 @BornChanger

2.8 Bug fixes

- TiDB
  - Fix the issue that a deadlock might occur when the tidb_restricted_read_only variable is set to true #53822 #55373 @Defined2014
  - Fix the issue that TiDB does not wait for auto-commit transactions to complete during graceful shutdown #55464 @YangKeao
  - Fix the issue that reducing the value of tidb_ttl_delete_worker_count during TTL job execution makes the job fail to complete #55561 @lcwangchao
  - Fix the issue that if the index of a table contains generated columns, an Unknown column 'column_name' in 'expression' error might occur when collecting statistics for the table via the ANALYZE statement #55438 @hawkingrei
  - Deprecate unnecessary configurations related to statistics to reduce redundant code #55043 @Rustin170506
  - Fix the issue that TiDB might hang or return incorrect results when executing a query containing a correlated subquery and CTE #55551 @guo-shaoge
  - Fix the issue that disabling lite-init-stats might cause statistics to fail to load synchronously #54532 @hawkingrei

- Fix the issue that when an UPDATE or DELETE statement contains a recursive CTE, the statement might report an error or not take effect #55666 @time-and-fate
- Fix the issue that a SQL binding containing window functions might not take effect in some cases #55981 @winoros
- Fix the issue that statistics for string columns with non-binary collations might fail to load when initializing statistics #55684 @winoros
- Fix the issue that the optimizer incorrectly estimates the number of rows as 1 when accessing a unique index with the query condition column IS NULL #56116 @hawkingrei
- Fix the issue that the optimizer does not use the best multi-column statistics information for row count estimation when the query contains filter conditions like (... AND ...) OR (... AND ...) ... #54323 @time-and-fate
- Fix the issue that the read_from_storage hint might not take effect when the query has an available Index Merge execution plan #56217 @AilinKid
- Fix the data race issue in IndexNestedLoopHashJoin #49692 @solotzg
- Fix the issue that the SUB_PART value in the INFORMATION_SCHEMA.STATISTICS table is NULL #55812 @Defined2014
- Fix the issue that an error occurs when a DML statement contains nested generated columns #53967 @wjhuang2016
- Fix the issue that the integer type of data with minimum display length in the division operation might cause the division result to overflow #55837 @windtalker
- Fix the issue that the operator that follows the TopN operator can not trigger the fallback action when the memory limit is exceeded #56185 @xzhangxian1008
- Fix the issue that the ORDER BY column in the Sort operator is stuck if it contains a constant #55344 @xzhangxian1008
- Fix the issue that when adding an index, the 8223 (HY000) error occurs after killing the PD leader and the data in the table is inconsistent #55488 @tangenta

    - Fix the issue that too many DDL history jobs cause OOM when you request information about history DDL jobs #55711 @joccau
    - Fix the issue that executing IMPORT INTO is stuck when Global Sort is enabled and the Region size exceeds 96 MiB #55374 @lance6716
    - Fix the issue that executing IMPORT INTO on a temporary table causes TiDB to crash #55970 @D3Hunter
    - Fix the issue that adding a unique index causes the duplicate entry error #56161 @tangenta
    - Fix the issue that TiDB Lightning does not ingest all KV pairs when TiKV is down for more than 810 seconds, resulting in inconsistent data in the table #55808 @lance6716
    - Fix the issue that the CREATE TABLE LIKE statement can not be used for cached tables #56134 @tiancaiamao
    - Fix the confusing warning message for FORMAT() expressions in CTE #56198 @dveeden
    - Fix the issue that column type restrictions are inconsistent between CREATE TABLE and ALTER TABLE when creating a partitioned table #56094 @mjonss
    - Fix the incorrect time type in the INFORMATION_SCHEMA.RUNAWAY_WATCHES table #54770 @HuSharp
- TiKV
    - Fix the issue that prevents master key rotation when the master key is stored in a Key Management Service (KMS) #17410 @hhwyt
    - Fix a traffic control issue that might occur after deleting large tables or partitions #17304 @Connor1996
    - Fix the issue that TiKV might panic when a stale replica processes Raft snapshots, triggered by a slow split operation and immediate removal of the new replica #17469 @hbisheng
- TiFlash
    - Fix the issue that TiFlash fails to parse the table schema when the table contains Bit-type columns with a default value that contains invalid characters #9461 @Lloyd-Pottiger

- Fix the issue that TiFlash might panic due to spurious Region overlap check failures that occur when multiple Regions are concurrently applying snapshots #9329 @CalvinNeo
- Fix the issue that some JSON functions unsupported by TiFlash are pushed down to TiFlash #9444 @windtalker

- Tools
  - Backup & Restore (BR)
    - Fix the issue that the PITR checkpoint interval in monitoring abnormally increased when TiDB nodes stopped, which does not reflect the actual situation #42419 @YuJuncen
    - Fix the issue that backup tasks might get stuck if TiKV becomes unresponsive during the backup process #53480 @Leavrth
    - Fix the issue that BR logs might print sensitive credential information when log backup is enabled #55273 @RidRisR
    - Fix the issue that after a log backup PITR task fails and you stop it, the safepoints related to that task are not properly cleared in PD #17316 @Leavrth
  - TiDB Data Migration (DM)
    - Fix the issue that multiple DM-master nodes might simultaneously become leaders, leading to data inconsistency #11602 @GMHDBJD
    - Fix the issue that DM does not set the default database when processing the `ALTER DATABASE` statement, which causes a replication error #11503 @lance6716
  - TiDB Lightning
    - Fix the issue that TiDB Lightning reports a `verify allocator base failed` error when two instances simultaneously start parallel import tasks and are assigned the same task ID #55384 @ei-sugimoto

2.9 Contributors

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

- ei-sugimoto
- eltociear
- guoshouyan (First-time contributor)
- JackL9u
- kafka1991 (First-time contributor)
- qingfeng777
- samba-rgb (First-time contributor)
- SeaRise
- tuziemon (First-time contributor)
- xyproto (First-time contributor)

3 TiDB 8.3.0 Release Notes

Release date: August 22, 2024

TiDB version: 8.3.0

Quick access: Quick start

8.3.0 introduces the following key features and improvements:

Category	Feature/Enhancement	Description
Scalability and Performance	Global indexes for partitioned tables (experimental)	Global indexes can effectively improve the efficiency of retrieving non-partitioned columns, and remove the restriction that a unique key must contain the partition key. This feature extends the usage scenarios of TiDB partitioned tables and avoids some of the application modification work that might be required for data migration.
	Default pushdown of the Projection operator to the storage engine	Pushing the Projection operator down to the storage engine can distribute the load across storage nodes while reducing data transfer between nodes. This optimization helps to reduce the execution time for certain SQL queries and improves the overall database performance.
	Ignoring unnecessary columns when collecting statistics	Under the premise of ensuring that the optimizer can obtain the necessary information, TiDB speeds up statistics collection, improves the timeliness of statistics, and thus ensures that the optimal execution plan is selected, improving the performance of the cluster. Meanwhile,

Category	Feature/Enhancement	Description
		TiDB also reduces the system overhead and improves the resource utilization.
Reliability and Availability	Built-in virtual IP management in TiProxy	TiProxy introduces built-in virtual IP management. When configured, it supports automatic virtual IP switching without relying on external platforms or tools. This feature simplifies TiProxy deployment and reduces the complexity of the database access layer.

3.1 Feature details

3.1.1 Performance

- The optimizer allows pushing the Projection operator down to the storage engine by default #51876 @yibin87

  Pushing the Projection operator down to the storage engine reduces data transfer between the compute engine and the storage engine, thereby improving SQL execution performance. This is particularly effective for queries containing JSON query functions or JSON value attribute functions. Starting from v8.3.0, TiDB enables the Projection operator pushdown feature by default, by changing the default value of the system variable controlling this feature, tidb_opt_projection_push_down, from OFF to ON. When this feature is enabled, the optimizer automatically pushes eligible JSON query functions and JSON value attribute functions down to the storage engine.

  For more information, see documentation.

- Optimize batch processing strategy for KV (key-value) requests #55206 @zyguan

  TiDB fetches data by sending KV requests to TiKV. Batching and processing KV requests in bulk can significantly improve execution performance. Before v8.3.0, the batching strategy in TiDB is less efficient. Starting from v8.3.0, TiDB introduces several more efficient batching strategies in addition to the existing one. You can configure different batching strategies using the tikv-client.batch-policy configuration item to accommodate various workloads.

  For more information, see documentation.

- TiFlash introduces HashAgg aggregation calculation modes to improve the performance for high NDV data #9196 @guo-shaoge

  Before v8.3.0, TiFlash has low aggregation calculation efficiency during the first stage of HashAgg aggregation when handling data with high NDV (number of distinct values). Starting from v8.3.0, TiFlash introduces multiple HashAgg aggregation calculation modes to improve the aggregation performance for different data characteristics. To choose a desired HashAgg aggregation calculation mode, you can configure the tiflash_hashagg_preaggregation_mode system variable.

  For more information, see documentation.

- Ignore unnecessary columns when collecting statistics #53567 @hi-rustin

  When the optimizer generates an execution plan, it only needs statistics for some columns, such as columns in the filter conditions, columns in the join keys, and columns used for aggregation. Starting from v8.3.0, TiDB continuously observes the historical records of the columns used in SQL statements. By default, TiDB only collects statistics for columns with indexes and columns that are observed to require statistics collection. This speeds up the collection of statistics and avoids unnecessary resource consumption.

  When you upgrade your cluster from a version earlier than v8.3.0 to v8.3.0 or later, TiDB retains the original behavior by default, that is, collecting statistics for all columns. To enable this feature, you need to manually set the system variable tidb_analyze_column_options to PREDICATE. For newly deployed clusters, this feature is enabled by default.

  For analytical systems with many random queries, you can set the system variable tidb_analyze_column_options to ALL to collect statistics for all columns, to ensure the performance of random queries. For other types of systems, it is recommended to keep the default setting (PREDICATE) of tidb_analyze_column_options to collect statistics for only necessary columns.

  For more information, see documentation.

- Improve the query performance of some system tables #50305 @tangenta

  In previous versions, querying system tables has poor performance when the cluster size becomes large and there are a large number of tables.

  In v8.0.0, query performance is optimized for the following four system tables:

  - INFORMATION_SCHEMA.TABLES
  - INFORMATION_SCHEMA.STATISTICS
  - INFORMATION_SCHEMA.KEY_COLUMN_USAGE
  - INFORMATION_SCHEMA.REFERENTIAL_CONSTRAINTS

  In v8.3.0, the query performance is optimized for the following system tables, bringing a multi-fold performance improvement compared to v8.2.0.

  - INFORMATION_SCHEMA.CHECK_CONSTRAINTS
  - INFORMATION_SCHEMA.COLUMNS
  - INFORMATION_SCHEMA.PARTITIONS
  - INFORMATION_SCHEMA.SCHEMATA
  - INFORMATION_SCHEMA.SEQUENCES
  - INFORMATION_SCHEMA.TABLE_CONSTRAINTS
  - INFORMATION_SCHEMA.TIDB_CHECK_CONSTRAINTS
  - INFORMATION_SCHEMA.TiDB_INDEXES
  - INFORMATION_SCHEMA.TIDB_INDEX_USAGE
  - INFORMATION_SCHEMA.VIEWS
- Support partition pruning when partition expressions use the EXTRACT(YEAR_MONTH...) function to improve query performance #54209 @mjonss

  In previous versions, when partition expressions use the EXTRACT(YEAR_MONTH...) function, partition pruning is not supported, resulting in poor query performance. Starting from v8.3.0, partition pruning is supported when partition expressions use the EXTRACT(YEAR_MONTH...) function, which improves query performance.

  For more information, see documentation.

- Improve the performance of CREATE TABLE by 1.4 times, CREATE DATABASE by 2.1 times, and ADD COLUMN by 2 times #54436 @D3Hunter

  TiDB v8.0.0 introduces the system variable tidb_enable_fast_create_table to improve table creation performance in batch table creation scenarios. In v8.3.0, when submitting the DDL statements for table creation concurrently through 10 sessions in a single database, the performance is improved by 1.4 times compared with v8.2.0.

  In v8.3.0, the performance of general DDLs in batch execution has improved compared to v8.2.0. The performance of CREATE DATABASE for 10 sessions concurrently improves by 19 times compared with v8.1.0 and 2.1 times compared with v8.2.0. The performance of using 10 sessions to add columns (ADD COLUMN) to multiple tables in the same database in batch has improved by 10 times compared with v8.1.0, and 2.1 times compared with v8.2.0. The performance of ADD COLUMN with 10 sessions on multiple tables in the same database has improved by 10 times compared with v8.1.0 and 2 times compared with v8.2.0.

  For more information, see documentation.

- Partitioned tables support global indexes (experimental) #45133 @mjonss @Defined2014 @jiyfhust @L-maple

  In previous versions of partitioned tables, some limitations exist because global indexes are not supported. For example, the unique key must use every column in the table's partitioning expression. If the query condition does not use the partitioning key, the query will scan all partitions, resulting in poor performance. Starting from v7.6.0, the system variable tidb_enable_global_index is introduced to enable the global index feature. But this feature was under development at that time and it is not recommended to enable it.

  Starting with v8.3.0, the global index feature is released as an experimental feature. You can explicitly create a global index for a partitioned table with the keyword Global to remove the restriction that the unique key must use every column in the table's partitioning expression, to meet flexible business needs. Global indexes also

enhance the performance of queries that do not include partition keys.

For more information, see documentation.

3.1.2 Reliability

- Support streaming cursor result sets (experimental) #54526 @YangKeao

  When the application code retrieves the result set using Cursor Fetch, TiDB usually first stores the complete result set in memory, and then returns the data to the client in batches. If the result set is too large, TiDB might temporarily write the result to the hard disk.

  Starting from v8.3.0, if you set the system variable tidb_enable_lazy_cursor_fetch to ON, TiDB no longer reads all data to the TiDB node, but gradually reads data to the TiDB node as the client reads. When TiDB processes large result sets, this feature reduces the memory usage of the TiDB node and improves the stability of the cluster.

  For more information, see documentation.

- Enhance SQL execution plan binding #55280 #55343 @time-and-fate

  In OLTP scenarios, the optimal execution plan for most SQL statements is fixed. Implementing SQL execution plan binding for important SQL statements in the application can reduce the probability of the execution plan becoming worse and improve system stability. To meet the requirements of creating a large number of SQL execution plan bindings, TiDB enhances the capability and experience of SQL binding, including:

  - Use a single SQL statement to create SQL execution plan bindings from multiple historical execution plans to improve the efficiency of creating bindings.
  - The SQL execution plan binding supports more optimizer hints, and optimizes the conversion method for complex execution plans, making the binding more stable in restoring the execution plan.

  For more information, see documentation.

3.1.3 Availability

- TiProxy supports built-in virtual IP management #583 @djshow832

  Before v8.3.0, when using primary-secondary mode for high availability, TiProxy requires an additional component to manage the virtual IP address. Starting from v8.3.0, TiProxy supports built-in virtual IP management. In primary-secondary mode, when a primary node fails over, the new primary node will automatically bind to the specified virtual IP, ensuring that clients can always connect to an available TiProxy through the virtual IP.

  To enable virtual IP management, specify the virtual IP address using the TiProxy configuration item ha.virtual-ip and specify the network interface to bind the virtual IP to using ha.interface. The virtual IP will be bound to a TiProxy instance only when both of these configuration items are set.

  For more information, see documentation.

3.1.4 SQL

- Support upgrading SELECT LOCK IN SHARE MODE to exclusive locks #54999 @cfzjywxk

  TiDB does not support SELECT LOCK IN SHARE MODE yet. Starting from v8.3.0, TiDB supports upgrading SELECT LOCK IN SHARE MODE to exclusive locks to enable support for SELECT LOCK IN SHARE MODE. You can control whether to enable this feature by using the new system variable tidb_enable_shared_lock_promotion.

  For more information, see documentation.

3.1.5 Observability

- Show the progress of loading initial statistics #53564 @hawkingrei

  TiDB loads basic statistics when it starts. In scenarios with many tables or partitions, this process can take a long time. When the configuration item force-init-stats is set to ON, TiDB does not provide services until the initial statistics are loaded. In this case, you need to observe the loading process to estimate the service start time.

  Starting from v8.3.0, TiDB prints the progress of loading initial statistics in stages in the log, so you can understand the running

status. To provide formatted results to external tools, TiDB adds the additional monitoring API so you can obtain the progress of loading initial statistics at any time during the startup phase.

- Add metrics about Request Unit (RU) settings #8444 @nolouch

3.1.6 Security

- Enhance PD log redaction #8305 @JmPotato

  TiDB v8.0.0 enhances log redaction and supports marking user data in TiDB logs with ‹ ›. Based on the marked logs, you can decide whether to redact the marked information when displaying the logs, thus increasing the flexibility of log redaction. In v8.2.0, TiFlash implements a similar log redaction enhancement.

  In v8.3.0, PD implements a similar log redaction enhancement. To use this feature, you can set the value of the PD configuration item security.redact-info-log to "marker".

  For more information, see documentation.

- Enhance TiKV log redaction #17206 @lucasliang

  TiDB v8.0.0 enhances log redaction and supports marking user data in TiDB logs with ‹ ›. Based on the marked logs, you can decide whether to redact the marked information when displaying the logs, thus increasing the flexibility of log redaction. In v8.2.0, TiFlash implements a similar log redaction enhancement.

  In v8.3.0, TiKV implements a similar log redaction enhancement. To use this feature, you can set the value of the TiKV configuration item security.redact-info-log to "marker".

  For more information, see documentation.

3.1.7 Data migration

- TiCDC supports replicating DDL statements in bi-directional replication (BDR) mode (GA) #10301 #48519 @okJiang @asddongmen

  TiCDC v7.6.0 introduced the replication of DDL statements with bi-directional replication configured. Previously, bi-directional replication of DDL statements was not supported by TiCDC, so users of TiCDC's bi-directional replication had to execute DDL statements on both TiDB

clusters separately. With this feature, after assigning a PRIMARY BDR role to a cluster, TiCDC can replicate the DDL statements from that cluster to the SECONDARY cluster.

In v8.3.0, this feature becomes generally available (GA).

For more information, see documentation.

3.2 Compatibility changes

Note:

This section provides compatibility changes you need to know when you upgrade from v8.2.0 to the current version (v8.3.0). If you are upgrading from v8.1.0 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

- To avoid incorrect use of commands, pd-ctl cancels the prefix matching mechanism. For example, store remove-tombstone cannot be called via store remove #8413 @lhy1024

3.2.2 System variables

Variable name	Change type	Description
tidb_ddl_reorg_batch_size	Modified	Adds the SESSION scope.
tidb_ddl_reorg_worker_cnt	Modified	Adds the SESSION scope.
tidb_gc_concurrency	Modified	Starting from v8.3.0, this variable controls the number of concurrent threads

Variable name	Change type	Description
		during the Resolve Locks and Delete Range steps of the Garbage Collection (GC) process. Before v8.3.0, this variable only controls the number of threads during the Resolve Locks step.
tidb_low_resolution_tso	Modified	Adds the GLOBAL scope.
tidb_opt_projection_push_down	Modified	Adds the GLOBAL scope and persists the variable value to the cluster. Changes the default value from OFF

Variable name	Change type	Description
		to ON after further tests, which means that the optimizer is allowed to push Projection down to the TiKV coprocessor.
tidb_schema_cache_size	Modified	The range of values has been modified to either 0 or [536870912, 9223372036854775807]. The minimum value is 536870912 bytes (that is, 512 MiB) to avoid setting a cache size that is too small and causing performance degradation.

Variable name	Change type	Description
tidb_analyze_column_options	Newly added	Controls the behavior of the ANALYZE TABLE statement. Setting it to the default value PREDICATE means only collecting statistics for predicate columns; setting it to ALL means collecting statistics for all columns.
tidb_enable_lazy_cursor_fetch	Newly added	Controls the behavior of the Cursor Fetch feature.
tidb_enable_shared_lock_promotion	Newly added	Controls whether to enable the feature of upgrading shared locks to exclusive locks. The

Variable name	Change type	Description
		default value of this variable is OFF, which means that the function of upgrading shared locks to exclusive locks is disabled.
tiflash_hashagg_preaggregation_mode	Newly added	Controls the pre-aggregation strategy used during the first stage of two-stage or three-stage HashAgg operations pushed down to TiFlash.

3.2.3 Configuration file parameters

Configuration file	Configuration parameter	Change type	Description
TiDB	tikv-client.batch-policy	Newly added	Controls the batching strategy for requests from TiDB to TiKV.
PD	security.redact-info-log	Modified	Support setting the value of the PD configuration item security.redact-info-log to "marker" to mark

Configuration file	Configuration parameter	Change type	Description
			sensitive information in the log with ‹ › instead of shielding it directly. With the "marker" option, you can customize the redaction rules.
TiKV	security.redact-info-log	Modified	Support setting the value of the TiKV configuration item security.redact-info-log to "marker" to mark sensitive information in the log with ‹ › instead of shielding it directly. With the "marker" option, you can customize the redaction rules.
TiFlash	security.redact-info-log	Modified	Support setting the value of the TiFlash Learner configuration item security.redact-info-log to "marker" to mark sensitive information in the log with ‹ › instead of shielding it directly. With the "marker" option, you can customize the redaction rules.
BR	--allow-pitr-from-incremental	Newly added	Controls whether incremental backups are compatible with subsequent log backups. The default value is true, which means that incremental backups are compatible with subsequent log backups. When you keep the default value true, the DDLs that need to be replayed are strictly checked before the incremental restore begins.

3.2.4 System tables

- The INFORMATION_SCHEMA.PROCESSLIST and INFORMATION_SCHEMA.CLUSTER_PROCESSLIST system tables add the SESSION_ALIAS field to show the number of rows currently affected by the DML statement #46889 @lcwangchao

3.3 Deprecated features

- The following features are deprecated starting from v8.3.0:
    - Starting from v7.5.0, TiDB Binlog replication is deprecated. Starting from v8.3.0, TiDB Binlog is fully deprecated, with removal planned for a future release. For incremental data replication, use TiCDC instead. For point-in-time recovery (PITR), use PITR.
    - Starting from v8.3.0, the tidb_enable_column_tracking system variable is deprecated. TiDB tracks predicate columns by default. For more information, see tidb_analyze_column_options.
- The following features are planned for deprecation in future versions:
    - TiDB introduces the system variable tidb_enable_auto_analyze_priority_queue, which controls whether priority queues are enabled to optimize the ordering of tasks that automatically collect statistics. In future releases, the priority queue will be the only way to order tasks for automatically collecting statistics, so this system variable will be deprecated.
    - TiDB introduces the system variable tidb_enable_async_merge_global_stats in v7.5.0. You can use it to set TiDB to use asynchronous merging of partition statistics to avoid OOM issues. In future releases, partition statistics will be merged asynchronously, so this system variable will be deprecated.
    - It is planned to redesign the automatic evolution of execution plan bindings in subsequent releases, and the related variables and behavior will change.
    - In v8.0.0, TiDB introduces the tidb_enable_parallel_hashagg_spill system variable to control whether TiDB supports disk spill for the concurrent HashAgg algorithm. In future versions, the

tidb_enable_parallel_hashagg_spill system variable will be deprecated.
    - The TiDB Lightning parameter conflict.max-record-rows is planned for deprecation in a future release and will be subsequently removed. This parameter will be replaced by conflict.threshold, which means that the maximum number of conflicting records is consistent with the maximum number of conflicting records that can be tolerated in a single import task.
- The following features are planned for removal in future versions:
    - Starting from v8.0.0, TiDB Lightning deprecates the old version of conflict detection strategy for the physical import mode, and enables you to control the conflict detection strategy for both logical and physical import modes via the conflict.strategy parameter. The duplicate-resolution parameter for the old version of conflict detection will be removed in a future release.

3.4 Improvements

- TiDB
    - Support the SELECT ... STRAIGHT_JOIN ... USING (...) statement #54162 @dveeden
    - Construct more precise index access ranges for filter conditions like ((idx_col_1 > 1) or (idx_col_1 = 1 and idx_col_2 > 10)) and ((idx_col_1 < 10) or (idx_col_1 = 10 and idx_col_2 < 20)) #54337 @ghazalfamilyusa
    - Use index order to avoid extra sorting operations for SQL queries like WHERE idx_col_1 IS NULL ORDER BY idx_col_2 #54188 @ari-e
    - Display analyzed indexes in the mysql.analyze_jobs system table #53567 @hi-rustin
    - Support applying the tidb_redact_log setting to the output of EXPLAIN statements and further optimize the logic in processing logs #54565 @hawkingrei
    - Support generating the Selection operator on IndexRangeScan for multi-valued indexes to improve query efficiency #54876 @time-and-fate
    - Support killing automatic ANALYZE tasks that are running outside the set time window #55283 @hawkingrei

- Adjust estimation results from 0 to 1 for equality conditions that do not hit TopN when statistics are entirely composed of TopN and the modified row count in the corresponding table statistics is non-zero #47400 @terry1purcell
- The TopN operator supports disk spill #47733 @xzhangxian1008
- TiDB node supports executing queries with the WITH ROLLUP modifier and the GROUPING function #42631 @Arenatlx
- The system variable tidb_low_resolution_tso supports the GLOBAL scope #55022 @cfzjywxk
- Improve GC (Garbage Collection) efficiency by supporting concurrent range deletion. You can control the number of concurrent threads using tidb_gc_concurrency #54570 @ekexium
- Improve the performance of bulk DML execution mode (tidb_dml_type = "bulk") #50215 @ekexium
- Improve the performance of schema information cache-related interface SchemaByID #54074 @ywqzzy
- Improve the query performance for certain system tables when schema information caching is enabled #50305 @tangenta
- Optimize error messages for conflicting keys when adding unique indexes #53004 @lance6716

- PD

  - Support modifying the batch configuration of the evict-leader-scheduler via pd-ctl to accelerate the leader eviction process #8265 @rleungx
  - Add the store_id monitoring metric to the **Cluster > Label distribution** panel in Grafana to display store IDs corresponding to different labels #8337 @HuSharp
  - Support fallback to the default resource group when the specified resource group does not exist #8388 @JmPotato
  - Add the approximate_kv_size field to the Region information output by the region command in pd-ctl #8412 @zeminzhou
  - Optimize the message that returns when you call the PD API to delete the TTL configuration #8450 @lhy1024
  - Optimize the RU consumption behavior of large query read requests to reduce the impact on other requests #8457 @nolouch

    - Optimize the error message that returns when you misconfigure PD microservices #52912 @rleungx
    - Add the --name startup parameter to PD microservices to more accurately display the service name during deployment #7995 @HuSharp
    - Support dynamically adjusting PatrolRegionScanLimit based on the number of Regions to reduce Region scan time #7963 @lhy1024
- TiKV
    - Optimize the batching policy for writing Raft logs when async-io is enabled to reduce the consumption of disk I/O bandwidth resources #16907 @LykxSassinator
    - Redesign the TiCDC delegate and downstream modules to better support Region partial subscription #16362 @hicqu
    - Reduce the size of a single slow query log #17294 @Connor1996
    - Add a new monitoring metric min safe ts #17307 @mittalrishabh
    - Reduce the memory usage of the peer message channel #16229 @Connor1996
- TiFlash
    - Support generating ad hoc heap profiling in SVG format #9320 @CalvinNeo
- Tools
    - Backup & Restore (BR)
        - Support checking whether a full backup exists before starting point-in-time recovery (PITR) for the first time. If the full backup is not found, BR terminates the restore and returns an error #54418 @Leavrth
        - Support checking whether the disk space in TiKV and TiFlash is sufficient before restoring snapshot backups. If the space is insufficient, BR terminates the restore and returns an error #54316 @RidRisR
        - Support checking whether the disk space in TiKV is sufficient before TiKV downloads each SST file. If the

space is insufficient, BR terminates the restore and returns an error #17224 @RidRisR
    - Support setting Alibaba Cloud access credentials through environment variables #45551 @RidRisR
    - Automatically set the environment variable GOMEMLIMIT based on the available memory of the BR process to avoid OOM when using BR for backup and restore #53777 @Leavrth
    - Make incremental backups compatible with point-in-time recovery (PITR) #54474 @3pointer
    - Support backing up and restoring the mysql.column_stats_usage table #53567 @hi-rustin

3.5 Bug fixes

- TiDB
    - Reset the parameters in the Open method of PipelinedWindow to fix the unexpected error that occurs when the PipelinedWindow is used as a child node of Apply due to the reuse of previous parameter values caused by repeated opening and closing operations #53600 @XuHuaiyu
    - Fix the issue that the query might get stuck when terminated because the memory usage exceeds the limit set by tidb_mem_quota_query #55042 @yibin87
    - Fix the issue that the disk spill for the HashAgg operator causes incorrect query results during parallel calculation #55290 @xzhangxian1008
    - Fix the issue of wrong JSON_TYPE when casting YEAR to JSON format #54494 @YangKeao
    - Fix the issue that the value range of the tidb_schema_cache_size system variable is wrong #54034 @lilinghai
    - Fix the issue that partition pruning does not work when the partition expression is EXTRACT(YEAR FROM col) #54210 @mjonss
    - Fix the issue that FLASHBACK DATABASE fails when many tables exist in the database #54415 @lance6716
    - Fix the issue that FLASHBACK DATABASE enters an infinite loop when handling many databases #54915 @lance6716
    - Fix the issue that adding an index in index acceleration mode might fail #54568 @lance6716

- Fix the issue that ADMIN CANCEL DDL JOBS might cause DDL to fail #54687 @lance6716
- Fix the issue that table replication fails when the index length of the table replicated from DM exceeds the maximum length specified by max-index-length #55138 @lance6716
- Fix the issue that the error runtime error: index out of range might occur when executing SQL statements with tidb_enable_inl_join_inner_multi_pattern enabled #54535 @joechenrh
- Fix the issue that you cannot exit TiDB using Control+C during the process of initializing statistics #54589 @tiancaiamao
- Fix the issue that the INL_MERGE_JOIN optimizer hint returns incorrect results by deprecating it #54064 @AilinKid
- Fix the issue that a correlated subquery that contains WITH ROLLUP might cause TiDB to panic and return the error runtime error: index out of range #54983 @AilinKid
- Fix the issue that predicates cannot be pushed down properly when the filter condition of a SQL query contains virtual columns and the execution condition contains UnionScan #54870 @qw4990
- Fix the issue that the error runtime error: invalid memory address or nil pointer dereference might occur when executing SQL statements with tidb_enable_inl_join_inner_multi_pattern enabled #55169 @hawkingrei
- Fix the issue that a query statement that contains UNION might return incorrect results #52985 @XuHuaiyu
- Fix the issue that the tot_col_size column in the mysql.stats_histograms table might be a negative number #55126 @qw4990
- Fix the issue that columnEvaluator cannot identify the column references in the input chunk, which leads to runtime error: index out of range when executing SQL statements #53713 @AilinKid
- Fix the issue that STATS_EXTENDED becomes a reserved keyword #39573 @wddevries
- Fix the issue that when tidb_low_resolution is enabled, select for update can be executed #54684 @cfzjywxk

- Fix the issue that internal SQL queries cannot be displayed in the slow query log when tidb_redact_log is enabled #54190 @lcwangchao
- Fix the issue that the memory used by transactions might be tracked multiple times #53984 @ekexium
- Fix the issue that using SHOW WARNINGS; to obtain warnings might cause a panic #48756 @xhebox
- Fix the issue that loading index statistics might cause memory leaks #54022 @hi-rustin
- Fix the issue that the LENGTH() condition is unexpectedly removed when the collation is utf8_bin or utf8mb4_bin #53730 @elsa0520
- Fix the issue that statistics collection does not update the stats_history table when encountering duplicate primary keys #47539 @Defined2014
- Fix the issue that recursive CTE queries might result in invalid pointers #54449 @hawkingrei
- Fix the issue that the Connection Count monitoring metric in Grafana is incorrect when some connections exit before the handshake is complete #54428 @YangKeao
- Fix the issue that the Connection Count of each resource group is incorrect when using TiProxy and resource groups #54545 @YangKeao
- Fix the issue that when queries contain non-correlated subqueries and LIMIT clauses, column pruning might be incomplete, resulting in a less optimal plan #54213 @qw4990
- Fix the issue of reusing wrong point get plans for SELECT ... FOR UPDATE #54652 @qw4990
- Fix the issue that the TIMESTAMPADD() function goes into an infinite loop when the first argument is month and the second argument is negative #54908 @xzhangxian1008
- Fix the issue that internal SQL statements in the slow log are redacted to null by default #54190 #52743 #53264 @lcwangchao
- Fix the issue that PointGet execution plans for _tidb_rowid can be generated #54583 @Defined2014
- Fix the issue that SHOW IMPORT JOBS reports an error Unknown column 'summary' after upgrading from v7.1 #54241 @tangenta

- Fix the issue that obtaining the column information using information_schema.columns returns warning 1356 when a subquery is used as a column definition in a view definition #54343 @lance6716
- Fix the issue that RANGE partitioned tables that are not strictly self-incrementing can be created #54829 @Defined2014
- Fix the issue that INDEX_HASH_JOIN cannot exit properly when SQL is abnormally interrupted #54688 @wshwsh12
- Fix the issue that the network partition during adding indexes using the Distributed eXecution Framework (DXF) might cause inconsistent data indexes #54897 @tangenta

- PD

  - Fix the issue that no error is reported when binding a role to a resource group #54417 @JmPotato
  - Fix the issue that a resource group encounters quota limits when requesting tokens for more than 500 ms #8349 @nolouch
  - Fix the issue that the time data type in the INFORMATION_SCHEMA.RUNAWAY_WATCHES table is incorrect #54770 @HuSharp
  - Fix the issue that resource groups could not effectively limit resource usage under high concurrency #8435 @nolouch
  - Fix the issue that an incorrect PD API is called when you retrieve table attributes #55188 @JmPotato
  - Fix the issue that the scaling progress is displayed incorrectly after the scheduling microservice is enabled #8331 @rleungx
  - Fix the issue that the encryption manager is not initialized before use #8384 @rleungx
  - Fix the issue that some logs are not redacted #8419 @rleungx
  - Fix the issue that redirection might panic during the startup of PD microservices #8406 @HuSharp
  - Fix the issue that the split-merge-interval configuration item might not take effect when you modify its value repeatedly (such as changing it from 1s to 1h and back to 1s) #8404 @lhy1024
  - Fix the issue that setting replication.strictly-match-label to true causes TiFlash to fail to start #8480 @rleungx

- Fix the issue that fetching TSO is slow when analyzing large partitioned tables, causing ANALYZE performance degradation #8500 @rleungx
- Fix the potential data races in large clusters #8386 @rleungx
- Fix the issue that when determining whether queries are Runaway Queries, TiDB only counts time consumption spent on the Coprocessor side while missing time consumption spent on the TiDB side, resulting in some queries not being identified as Runaway Queries #51325 @HuSharp

- TiFlash
  - Fix the issue that when using the CAST() function to convert a string to a datetime with a time zone or invalid characters, the result is incorrect #8754 @solotzg
  - Fix the issue that TiFlash might panic after executing RENAME TABLE ... TO ... on a partitioned table with empty partitions across databases #9132 @JaySon-Huang
  - Fix the issue that some queries might report a column type mismatch error after late materialization is enabled #9175 @JinheLin
  - Fix the issue that queries with virtual generated columns might return incorrect results after late materialization is enabled #9188 @JinheLin
  - Fix the issue that setting the SSL certificate configuration to an empty string in TiFlash incorrectly enables TLS and causes TiFlash to fail to start #9235 @JaySon-Huang
  - Fix the issue that TiFlash might panic when a database is deleted shortly after creation #9266 @JaySon-Huang
  - Fix the issue that a network partition (network disconnection) between TiFlash and any PD might cause read request timeout errors #9243 @Lloyd-Pottiger
  - Fix the issue that TiFlash write nodes might fail to restart in the disaggregated storage and compute architecture #9282 @JaySon-Huang
  - Fix the issue that read snapshots of TiFlash write nodes are not released in a timely manner in the disaggregated storage and compute architecture #9298 @JinheLin
- TiKV

- Fix the issue that cleaning up stale regions might accidentally delete valid data #17258 @hbisheng
- Fix the issue that Ingestion picked level and Compaction Job Size(files) are displayed incorrectly in the TiKV dashboard in Grafana #15990 @Connor1996
- Fix the issue that cancel_generating_snap incorrectly updating snap_tried_cnt causes TiKV to panic #17226 @hbisheng
- Fix the issue that the information of Ingest SST duration seconds is incorrect #17239 @LykxSassinator
- Fix the issue that CPU profiling flag is not reset correctly when an error occurs #17234 @Connor1996
- Fix the issue that bloom filters are incompatible between earlier versions (earlier than v7.1) and later versions #17272 @v01dstar

- Tools
  - Backup & Restore (BR)
    - Fix the issue that DDLs requiring backfilling, such as ADD INDEX and MODIFY COLUMN, might not be correctly recovered during incremental restore #54426 @3pointer
    - Fix the issue that the progress is stuck during backup and restore #54140 @Leavrth
    - Fix the issue that the checkpoint path of backup and restore is incompatible with some external storage #55265 @Leavrth
  - TiCDC
    - Fix the issue that the processor might get stuck when the downstream Kafka is inaccessible #11340 @asddongmen
  - TiDB Data Migration (DM)
    - Fix the issue that schema tracker incorrectly handles LIST partition tables, causing DM errors #11408 @lance6716
    - Fix the issue that data replication is interrupted when the index length exceeds the default value of max-index-length #11459 @michaelmdeng
    - Fix the issue that DM cannot handle FAKE_ROTATE_EVENT correctly #11381 @lance6716
  - TiDB Lightning

- Fix the issue that TiDB Lightning outputs a confusing WARN log when it fails to obtain the keyspace name #54232 @kennytm
- Fix the issue that the TLS configuration of TiDB Lightning affects cluster certificates #54172 @ei-sugimoto
- Fix the issue that transaction conflicts occur during data import using TiDB Lightning #49826 @lance6716
- Fix the issue that large checkpoint files cause performance degradation during the import of numerous databases and tables #55054 @D3Hunter

3.6 Contributors

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

- ari-e
- ei-sugimoto
- HaoW30
- JackL9u
- michaelmdeng
- mittalrishabh
- qingfeng777
- SandeepPadhi
- yzhan1

4 TiDB 8.2.0 Release Notes

Release date: July 11, 2024

TiDB version: 8.2.0

Quick access: Quick start

8.2.0 introduces the following key features and improvements:

Category	Feature/Enhancement	Description
Reliability and Availability	TiProxy supports multiple load balancing policies	In TiDB v8.2.0, TiProxy evaluates and ranks TiDB nodes based on various dimensions, such as status, connection counts, health, memory, CPU, and location. According to the load balancing policy specified in the policy configuration item, TiProxy dynamically selects the optimal TiDB node to execute

Category	Feature/Enhancement	Description
		database operations. This optimizes overall resource usage, improves cluster performance, and increases throughput.
	The parallel HashAgg algorithm of TiDB supports disk spill (GA)	HashAgg is a widely used aggregation operator in TiDB for efficiently aggregating rows with the same field values. TiDB v8.0.0 introduces parallel HashAgg as an experimental feature to further enhance processing speed. When memory resources are insufficient, parallel HashAgg spills temporary sorted data to disk, avoiding potential OOM risks caused by excessive memory usage. This improves query performance while maintaining node stability. In v8.2.0, this feature becomes generally available (GA) and is enabled by default, enabling you to safely configure the concurrency of parallel HashAgg using tidb_executor_concurrency.
	Improve statistics loading efficiency by up to 10 times	For clusters with a large number of tables and partitions, such as SaaS or PaaS services, improvement in statistics loading efficiency can solve the problem of slow startup of TiDB instances, and increase the success rate of dynamic loading of statistics. This improvement reduces performance rollbacks caused by statistics loading failures and improves cluster stability.
DB Operations and Observability	Introduce privilege control of switching resource groups	As resource control is widely used, the privilege control of switching resource groups can prevent database users from abusing resources, strengthen administrators' protection of overall resource usage, and improve cluster stability.

4.1 Feature details

4.1.1 Performance

- Support pushing down the following JSON functions to TiKV #50601 @dbsid
    - JSON_ARRAY_APPEND()
    - JSON_MERGE_PATCH()
    - JSON_REPLACE()

For more information, see documentation.

- TiDB supports parallel sorting #49217 #50746 @xzhangxian1008

  Before v8.2.0, TiDB only executes Sort operators sequentially, affecting query performance when sorting large amounts of data.

  Starting from v8.2.0, TiDB supports parallel sorting, which significantly improves sorting performance. This feature does not need manual configuration. TiDB automatically determines whether to use parallel sorting based on the value of the tidb_executor_concurrency system variable.

  For more information, see documentation.

- The parallel HashAgg algorithm of TiDB supports disk spill (GA) #35637 @xzhangxian1008

  TiDB v8.0.0 introduces the parallel HashAgg algorithm with disk spill support as an experimental feature. In v8.2.0, this feature becomes generally available (GA). When using the parallel HashAgg algorithm, TiDB automatically triggers data spill based on memory usage, thus balancing query performance and data throughput. This feature is enabled by default. The system variable tidb_enable_parallel_hashagg_spill, which controls this feature, will be deprecated in a future release.

  For more information, see documentation.

4.1.2 Reliability

- Improve statistics loading efficiency by up to 10 times #52831 @hawkingrei

  SaaS or PaaS applications can have a large number of data tables, which not only slow down the loading speed of the initial statistics, but also increase the failure rate of load synchronization under high loads. The startup time of TiDB and the accuracy of the execution plan can be affected. In v8.2.0, TiDB optimizes the process of loading statistics from multiple perspectives, such as the concurrency model and memory allocation, to reduce latency, improve throughput, and avoid slow loading of statistics that affect business scaling.

Adaptive concurrent loading is now supported. By default, the configuration item stats-load-concurrency is set to 0, and the concurrency of statistics loading is automatically selected based on the hardware specification.

For more information, see documentation.

4.1.3 Availability

- TiProxy supports multiple load balancing policies #465 @djshow832 @xhebox

  TiProxy is the official proxy component of TiDB, located between the client and TiDB server. It provides load balancing and connection persistence functions for TiDB. Before v8.2.0, TiProxy defaults to v1.0.0, which only supports status-based and connection count-based load balancing policies for TiDB servers.

  Starting from v8.2.0, TiProxy defaults to v1.1.0 and introduces multiple load balancing policies. In addition to status-based and connection count-based policies, TiProxy supports dynamic load balancing based on health, memory, CPU, and location, improving the stability of the TiDB cluster.

  You can configure the combination and priority of load balancing policies through the policy configuration item.

  - resource: the resource priority policy performs load balancing based on the following priority order: status, health, memory, CPU, location, and connection count.
  - location: the location priority policy performs load balancing based on the following priority order: status, location, health, memory, CPU, and connection count.
  - connection: the minimum connection count priority policy performs load balancing based on the following priority order: status and connection count.

  For more information, see documentation.

4.1.4 SQL

- TiDB supports the JSON schema validation function #52779 @dveeden

Before v8.2.0, you need to rely on external tools or customized validation logic for JSON data validation, which increases the complexity of development and maintenance, and reduces development efficiency. Starting from v8.2.0, the JSON_SCHEMA_VALID() function is introduced. Using JSON_SCHEMA_VALID() in the CHECK constraint can help prevent non-conforming data from being inserted, rather than checking the data after it has been added. This function lets you verify the validity of JSON data directly in TiDB, improving the integrity and consistency of the data, and increasing the development efficiency.

For more information, see documentation.

4.1.5 DB operations

- TiUP supports deploying PD microservices #5766 @rleungx

  Starting from v8.0.0, PD supports the microservice mode. This mode splits the timestamp allocation and cluster scheduling functions of PD into separate microservices that can be deployed independently, thereby improving resource control and isolation, and reducing the impact between different services. Before v8.2.0, PD microservices can only be deployed using TiDB Operator.

  Starting from v8.2.0, PD microservices can also be deployed using TiUP. You can deploy the tso microservice and the scheduling microservice separately in a cluster to enhance PD performance scalability and address PD performance bottlenecks in large-scale clusters. It is recommended to use this mode when PD becomes a significant performance bottleneck that cannot be resolved by scaling up.

  For more information, see user documentation.

- Add privilege control of switching resource groups #53440 @glorv

  TiDB lets users switch to other resource groups using the SET RESOURCE GROUP command or the RESOURCE_GROUP() hint, which might lead to resource group abuse by some database users. TiDB v8.2.0 introduces privilege control of switching resource groups. Only database users granted the RESOURCE_GROUP_ADMIN or RESOURCE_GROUP_USER dynamic privilege can switch to other resource groups, enhancing the protection of system resources.

To maintain compatibility, the original behavior is retained when upgrading from earlier versions to v8.2.0 or later versions. To enable the enhanced privilege control, set the new variable tidb_resource_control_strict_mode to ON.

For more information, see user documentation.

4.1.6 Observability

- Record the reason why an execution plan is not cached #50618 @qw4990

  In some scenarios, you might want to cache most execution plans to save execution overhead and reduce latency. Currently, execution plan caching has some limitations on SQL. Execution plans of some SQL statements cannot be cached. It is difficult to identify the SQL statements that cannot be cached and the corresponding reasons.

  Therefore, starting from v8.2.0, new columns PLAN_CACHE_UNQUALIFIED and PLAN_CACHE_UNQUALIFIED_LAST_REASON are added to the system table STATEMENTS_SUMMARY to explain the reason why an execution plan cannot be cached, which can help you tune performance.

  For more information, see documentation.

4.1.7 Security

- Enhance TiFlash log desensitization #8977 @JaySon-Huang

  TiDB v8.0.0 enhances the log desensitization feature, enabling you to control whether user data in TiDB logs is wrapped in markers ‹ ›. Based on the marked logs, you can decide whether to redact the marked information when displaying logs, thereby increasing the flexibility of log desensitization. In v8.2.0, TiFlash introduces a similar enhancement for log desensitization. To use this feature, set the TiFlash configuration item security.redact_info_log to marker.

  For more information, see documentation.

4.1.8 Data migration

- Align TiCDC Syncpoints across multiple changefeeds #11212 @hongyunyan

Before v8.2.0, aligning TiCDC Syncpoints across multiple changefeeds was challenging. The startTs of the changefeed had to be carefully selected when the changefeed was created, so it would align with the Syncpoints of other changefeeds. Starting from v8.2.0, Syncpoints for a changefeed are created as a multiple of the changefeed's sync-point-interval configuration. This change lets you align Syncpoints across multiple changefeeds that have the same sync-point-interval configuration, simplifying and improving the ability to align multiple downstream clusters.

For more information, see documentation.

- TiCDC Pulsar Sink supports using the pulsar+http and pulsar+https connection protocols #11336 @SandeepPadhi

  Before v8.2.0, TiCDC Pulsar Sink only supports pulsar and pulsar+ssl connection protocols. Starting from v8.2.0, TiCDC Pulsar Sink also supports pulsar+http and pulsar+https protocols for connections. This enhancement improves the flexibility of connecting to Pulsar.

  For more information, see documentation.

4.2 Compatibility changes

Note:

This section provides compatibility changes you need to know when you upgrade from v8.1.0 to the current version (v8.2.0). If you are upgrading from v8.0.0 or earlier versions to the current version, you might also need to check the compatibility changes introduced in intermediate versions.

4.2.1 Behavior changes

- When using TiDB Lightning to import a CSV file, if you set strict-format = true to split a large CSV file into multiple small CSV files to improve concurrency and import performance, you need to explicitly specify terminator. The values can be \r, \n or \r\n. Failure to specify a line terminator might result in an exception when parsing the CSV file data. #37338 @lance6716

- When using IMPORT INTO to import a CSV file, if you specify the SPLIT_FILE parameter to split a large CSV file into multiple small CSV files to improve concurrency and import performance, you need to

explicitly specify the line terminator LINES_TERMINATED_BY. The values can be \r, \n or \r\n. Failure to specify a line terminator might result in an exception when parsing the CSV file data. #37338 @lance6716

- Before BR v8.2.0, performing BR data restore on a cluster with TiCDC replication tasks is not supported. Starting from v8.2.0, BR relaxes the restrictions on data restoration for TiCDC: if the BackupTS (the backup time) of the data to be restored is earlier than the changefeed CheckpointTS (the timestamp that indicates the current replication progress), BR can proceed with the data restore normally. Considering that BackupTS is usually much earlier, it can be assumed that in most scenarios, BR supports restoring data for a cluster with TiCDC replication tasks. #53131 @YuJuncen

4.2.2 MySQL compatibility

- Before v8.2.0, executing the CREATE USER statement with the PASSWORD REQUIRE CURRENT DEFAULT option returns an error because this option is not supported and cannot be parsed. Starting from v8.2.0, TiDB supports parsing and ignoring this option for compatibility with MySQL. #53305 @dveeden

4.2.3 System variables

Variable name	Change type	Description
tidb_analyze_distsql_scan_concurrency	Modified	Changes the minimum value from 1 to 0. When you set it to 0, TiDB adaptively adjusts the concurrency of the scan operation when executing the ANALYZE operation based on the cluster size.
tidb_analyze_skip_column_types	Modified	Starting from v8.2.0, TiDB does not collect columns of MEDIUMTEXT and LONGTEXT types by default to avoid potential OOM risks.
tidb_enable_historical_stats	Modified	Changes the default value from ON to OFF, which turns off historical statistics to avoid potential stability issues.
tidb_executor_concurrency	Modified	Adds support for setting the concurrency of the sort operator.
tidb_sysproc_scan_concurrency	Modified	Changes the minimum value from 1 to 0. When you set it to 0, TiDB adaptively adjusts the concurrency of scan operations

Variable name	Change type	Description
		performed when executing internal SQL statements based on the cluster size.
tidb_resource_control_strict_mode	Newly added	Controls whether privilege control is applied to the SET RESOURCE GROUP statement and the RESOURCE_GROUP() optimizer hint.

4.2.4 Configuration file parameters

Configuration file	Configuration parameter	Change type	Description
TiDB	stats-load-concurrency	Modified	Changes the default value from 5 to 0, and the minimum value from 1 to 0. The value 0 means the automatic mode, which automatically adjusts concurrency based on the configuration of the server.
TiDB	token-limit	Modified	Changes the maximum value from 18446744073709551615 (64-bit platform) and 4294967295 (32-bit platform) to 1048576 to avoid causing TiDB Server OOM when setting it too large. It means that the number of sessions that can execute requests concurrently can be configured to a maximum of 1048576.
TiKV	max-apply-unpersisted-log-limit	Modified	Changes the default value from 0 to 1024 to reduce long-tail latency caused by I/O jitter on the TiKV node. It means that the maximum number of committed but not persisted Raft logs that can be applied is 1024 by default.

Configuration file	Configuration parameter	Change type	Description
TiKV	server.grpc-compression-type	Modified	This configuration item now also controls the compression algorithm of response messages sent from TiKV to TiDB. Enabling compression might consume more CPU resources.
TiFlash	security.redact_info_log	Modified	Introduces a new value option marker. When you set the value to marker, all user data in the log is wrapped in ‹ ›.

4.2.5 System tables

- The INFORMATION_SCHEMA.PROCESSLIST and INFORMATION_SCHEMA.CLUSTER_PROCESSLIST system tables add the SESSION_ALIAS field to show the alias of the current session. #46889 @lcwangchao

4.2.6 Compiler versions

- To improve the TiFlash development experience, the minimum version of LLVM required to compile and build TiDB has been upgraded from 13.0 to 17.0. If you are a TiDB developer, you need to upgrade the version of your LLVM compiler to ensure a smooth build. #7193 @Lloyd-Pottiger

4.3 Deprecated features

- The following features are deprecated starting from v8.2.0:
    - Starting from v8.2.0, the enable-replica-selector-v2 configuration item is deprecated. The new version of the Region replica selector is used by default when sending RPC requests to TiKV.
    - Starting from v8.2.0, the BR snapshot restore parameter --concurrency is deprecated. As an alternative, you can configure the maximum number of concurrent tasks per TiKV node during snapshot restore using --tikv-max-restore-concurrency.
    - Starting from v8.2.0, the BR snapshot restore parameter --granularity is deprecated, and the coarse-grained Region scattering algorithm is enabled by default.

- The following features are planned for deprecation in future versions:
    - In v8.0.0, TiDB introduces the tidb_enable_auto_analyze_priority_queue system variable to control whether to enable the priority queue to optimize the ordering of automatic statistics collection tasks. In future versions, the priority queue will become the only way to order automatic statistics collection tasks, and the tidb_enable_auto_analyze_priority_queue system variable will be deprecated.
    - In v8.0.0, TiDB introduces the tidb_enable_parallel_hashagg_spill system variable to control whether TiDB supports disk spill for the concurrent HashAgg algorithm. In future versions, the tidb_enable_parallel_hashagg_spill system variable will be deprecated.
    - In v7.5.0, TiDB introduces the tidb_enable_async_merge_global_stats system variable to enable TiDB to merge partition statistics asynchronously to avoid OOM issues. In future versions, partition statistics will be merged asynchronously by default, and the tidb_enable_async_merge_global_stats system variable will be deprecated.
    - It is planned to redesign the auto-evolution of execution plan bindings in subsequent releases, and the related variables and behavior will change.
    - The TiDB Lightning parameter conflict.max-record-rows is planned for deprecation in a future release and will be subsequently removed. This parameter will be replaced by conflict.threshold, which means that the maximum number of conflicting records is consistent with the maximum number of conflicting records that can be tolerated in a single import task.
- The following features are planned for removal in future versions:
    - Starting from v8.0.0, TiDB Lightning deprecates the old version of conflict detection strategy for the physical import mode, and enables you to control the conflict detection strategy for both logical and physical import modes via the conflict.strategy parameter. The duplicate-resolution parameter for the old version of conflict detection will be removed in a future release.

4.4 Improvements

- TiDB
    - Support parallel execution of logical DDL statements (General DDL). Compared with v8.1.0, when you use 10 sessions to submit different DDL statements concurrently, the performance is improved by 3 to 6 times #53246 @D3Hunter
    - Improve the logic of matching multi-column indexes using expressions like ((a = 1 and b = 2 and c > 3) or (a = 4 and b = 5 and c > 6)) and d > 3 to produce a more accurate Range #41598 @ghazalfamilyusa
    - Optimize the performance of obtaining data distribution information when performing simple queries on tables with large data volumes #53850 @you06
    - The aggregated result set can be used as an inner table for IndexJoin, allowing more complex queries to be matched to IndexJoin, thus improving query efficiency through indexing #37068 @elsa0520
    - By batch deleting TiFlash placement rules, improve the processing speed of data GC after performing the TRUNCATE or DROP operation on partitioned tables #54068 @Lloyd-Pottiger
    - Upgrade the version of Azure Identity Libraries and Microsoft Authentication Library to enhance security #53990 @hawkingrei
    - Set the maximum value of token-limit to 1048576 to avoid causing TiDB Server OOM when setting it too large #53312 @djshow832
    - Improve column pruning for MPP execution plans to improve TiFlash MPP execution performance #52133 @yibin87
    - Optimize the performance overhead of the IndexLookUp operator when looking up a table with a large amount of data (>1024 rows) #53871 @crazycs520
    - Remove stores without Regions during MPP load balancing #52313 @xzhangxian1008
- TiKV
    - Add the **Compaction Job Size(files)** metric to show the number of SST files involved in a single compaction job #16837 @zhangjinpeng87

    - Enable the early apply feature by default. With this feature enabled, the Raft leader can apply logs after quorum peers have persisted the logs, without waiting for the leader itself to persist the log, reducing the impact of jitter in a few TiKV nodes on write request latency #16717 @glorv
    - Improve the observability of **Raft dropped messages** to locate the root cause of slow writes #17093 @Connor1996
    - Improve the observability of ingest files latency to troubleshoot cluster latency issues #17078 @LykxSassinator
    - Use a separate thread to clean up Region replicas to ensure stable latency on critical Raft reads and writes #16001 @hbisheng
    - Improve the observability of the number of snapshots being applied #17078 @hbisheng
- PD
    - Improve the performance of Region heartbeat processing #7897 @nolouch @rleungx @JmPotato
    - pd-ctl supports querying hot Regions by the byte or query dimension #7369 @lhy1024
- TiFlash
    - Reduce lock conflicts under highly concurrent data read operations and optimize short query performance #9125 @JinheLin
    - Eliminate redundant copies of the Join Key in the Join operator #9057 @gengliqi
    - Concurrently perform the process of converting a two-level hash table in the HashAgg operator #8956 @gengliqi
    - Remove redundant aggregation functions for the HashAgg operator to reduce computational overhead #8891 @guo-shaoge
- Tools
    - Backup & Restore (BR)
        - Optimize the backup feature, improving backup performance and stability during node restarts, cluster

scaling-out, and network jitter when backing up large numbers of tables #52534 @3pointer

- Implement fine-grained checks of TiCDC changefeed during data restore. If the changefeed CheckpointTS is later than the data backup time, the restore operations are not affected, thereby reducing unnecessary wait times and improving user experience #53131 @YuJuncen
- Add several commonly used parameters to the BACKUP statement and the RESTORE statement, such as CHECKSUM_CONCURRENCY #53040 @RidRisR
- Except for the br log restore subcommand, all other br log subcommands support skipping the loading of the TiDB domain data structure to reduce memory consumption #52088 @Leavrth
- Support encryption of temporary files generated during log backup #15083 @YuJuncen
- Add a tikv_log_backup_pending_initial_scan monitoring metric in the Grafana dashboard #16656 @3pointer
- Optimize the output format of PITR logs and add a RestoreTS field in the logs #53645 @dveeden

- TiCDC
  - Support directly outputting raw events when the downstream is a Message Queue (MQ) or cloud storage #11211 @CharlesCheung96

4.5 Bug fixes

- TiDB
  - Fix the issue that when a SQL statement contains an Outer Join and the Join condition includes the false IN (column_name) expression, the query result lacks some data #49476 @ghazalfamilyusa
  - Fix the issue that statistics for columns in system tables are collected when TiDB collects PREDICATE COLUMNS statistics for tables #53403 @hi-rustin
  - Fix the issue that the tidb_enable_column_tracking system variable does not take effect when the

tidb_persist_analyze_options system variable is set to OFF #53478 @hi-rustin

- Fix the issue of potential data races during the execution of (*PointGetPlan).StatsInfo() #49803 #43339 @qw4990
- Fix the issue that TiDB might return incorrect query results when you query tables with virtual columns in transactions that involve data modification operations #53951 @qw4990
- Fix the issue that the tidb_enable_async_merge_global_stats and tidb_analyze_partition_concurrency system variables do not take effect during automatic statistics collection #53972 @hi-rustin
- Fix the issue that TiDB might return the plan not supported error when you query TABLESAMPLE #54015 @tangenta
- Fix the issue that executing the SELECT DISTINCT CAST(col AS DECIMAL), CAST(col AS SIGNED) FROM ... query might return incorrect results #53726 @hawkingrei
- Fix the issue that queries cannot be terminated after a data read timeout on the client side #44009 @wshwsh12
- Fix the overflow issue of the Longlong type in predicates #45783 @hawkingrei
- Fix the issue that the Window function might panic when there is a related subquery in it #42734 @hi-rustin
- Fix the issue that the TopN operator might be pushed down incorrectly #37986 @qw4990
- Fix the issue that SELECT INTO OUTFILE does not work when clustered indexes are used as predicates #42093 @qw4990
- Fix the issue that the query latency of stale reads increases, caused by information schema cache misses #53428 @crazycs520
- Fix the issue that comparing a column of YEAR type with an unsigned integer that is out of range causes incorrect results #50235 @qw4990
- Fix the issue that the histogram and TopN in the primary key column statistics are not loaded after restarting TiDB #37548 @hawkingrei
- Fix the issue that the final AggMode and the non-final AggMode cannot coexist in Massively Parallel Processing (MPP) #51362 @AilinKid

- Fix the issue that TiDB panics when executing the SHOW ERRORS statement with a predicate that is always true #46962 @elsa0520
- Fix the issue that using a view does not work in recursive CTE #49721 @hawkingrei
- Fix the issue that TiDB might report an error due to GC when loading statistics at startup #53592 @you06
- Fix the issue that PREPARE/EXECUTE statements with the CONV expression containing a ? argument might result in incorrect query results when executed multiple times #53505 @qw4990
- Fix the issue that non-BIGINT unsigned integers might produce incorrect results when compared with strings/decimals #41736 @LittleFall
- Fix the issue that TiDB does not create corresponding statistics metadata (stats_meta) when creating a table with foreign keys #53652 @hawkingrei
- Fix the issue that certain filter conditions in queries might cause the planner module to report an invalid memory address or nil pointer dereference error #53582 #53580 #53594 #53603 @YangKeao
- Fix the issue that executing CREATE OR REPLACE VIEW concurrently might result in the table doesn't exist error #53673 @tangenta
- Fix the issue that the STATE field in the INFORMATION_SCHEMA.TIDB_TRX table is empty due to the size of the STATE field not being defined #53026 @cfzjywxk
- Fix the issue that the Distinct_count information in global statistics might be incorrect when tidb_enable_async_merge_global_stats is disabled #53752 @hawkingrei
- Fix the issue of incorrect WARNINGS information when using Optimizer Hints #53767 @hawkingrei
- Fix the issue that negating a time type results in an incorrect value #52262 @solotzg
- Fix the issue that REGEXP() does not explicitly report an error for empty pattern arguments #53221 @yibin87
- Fix the issue that converting JSON to datetime might lose precision in some cases #53352 @YangKeao

- Fix the issue that JSON_QUOTE() returns incorrect results in some cases #37294 @dveeden
- Fix the issue that executing ALTER TABLE ... REMOVE PARTITIONING might cause data loss #53385 @mjonss
- Fix the issue that TiDB fails to reject unauthenticated user connections in some cases when using the auth_socket authentication plugin #54031 @lcwangchao
- Fix the issue that JSON-related functions return errors inconsistent with MySQL in some cases #53799 @dveeden
- Fix the issue that the INDEX_LENGTH field of partitioned tables in INFORMATION_SCHEMA.PARTITIONS is incorrect #54173 @Defined2014
- Fix the issue that the TIDB_ROW_ID_SHARDING_INFO field in the INFORMATION_SCHEMA.TABLES table is incorrect #52330 @tangenta
- Fix the issue that a generated column returns illegal timestamps #52509 @lcwangchao
- Fix the issue that setting max-index-length causes TiDB to panic when adding indexes using the Distributed eXecution Framework (DXF) #53281 @zimulala
- Fix the issue that the illegal column type DECIMAL(0,0) can be created in some cases #53779 @tangenta
- Fix the issue that using CURRENT_DATE() as the default value for a column results in incorrect query results #53746 @tangenta
- Fix the issue that the ALTER DATABASE ... SET TIFLASH REPLICA statement incorrectly adds TiFlash replicas to the SEQUENCE table #51990 @jiyfhust
- Fix the issue that the REFERENCED_TABLE_SCHEMA field in the INFORMATION_SCHEMA.KEY_COLUMN_USAGE table is incorrect #52350 @wd0517
- Fix the issue that inserting multiple rows in a single statement causes the AUTO_INCREMENT column to be discontinuous when AUTO_ID_CACHE=1 #52465 @tiancaiamao
- Fix the format of deprecation warnings #52515 @dveeden
- Fix the issue that the TRACE command is missing in copr.buildCopTasks #53085 @time-and-fate
- Fix the issue that the memory_quota hint might not work in subqueries #53834 @qw4990

    - Fix the issue that improper use of metadata locks might lead to writing anomalous data when using the plan cache under certain circumstances #53634 @zimulala
    - Fix the issue that after a statement within a transaction is killed by OOM, if TiDB continues to execute the next statement within the same transaction, you might get an error Trying to start aggressive locking while it's already started and a panic occurs #53540 @MyonKeminta
- TiKV
    - Fix the issue that pushing down the JSON_ARRAY_APPEND() function to TiKV causes TiKV to panic #16930 @dbsid
    - Fix the issue that the leader does not clean up failed snapshot files in time #16976 @hbisheng
    - Fix the issue that highly concurrent Coprocessor requests might cause TiKV OOM #16653 @overvenus
    - Fix the issue that changing the raftstore.periodic-full-compact-start-times configuration item online might cause TiKV to panic #17066 @SpadeA-Tang
    - Fix the failure of make docker and make docker_test #17075 @shunki-fujita
    - Fix the issue that the **gRPC request sources duration** metric is displayed incorrectly in the monitoring dashboard #17133 @King-Dylan
    - Fix the issue that setting the gRPC message compression method via grpc-compression-type does not take effect on messages sent from TiKV to TiDB #17176 @ekexium
    - Fix the issue that the output of the raft region command in tikv-ctl does not include the Region status information #17037 @glorv
    - Fix the issue that CDC and log-backup do not limit the timeout of check_leader using the advance-ts-interval configuration, causing the resolved_ts lag to be too large when TiKV restarts normally in some cases #17107 @MyonKeminta
- PD
    - Fix the issue that ALTER PLACEMENT POLICY cannot modify the placement policy #52257 #51712 @jiyfhust

    - Fix the issue that the scheduling of write hotspots might break placement policy constraints #7848 @lhy1024
    - Fix the issue that down peers might not recover when using Placement Rules #7808 @rleungx
    - Fix the issue that a large number of retries occur when canceling resource groups queries #8217 @nolouch
    - Fix the issue that manually transferring the PD leader might fail #8225 @HuSharp
- TiFlash
    - Fix the issue of query timeout when executing queries on partitioned tables that contain empty partitions #9024 @JinheLin
    - Fix the issue that in the disaggregated storage and compute architecture, null values might be incorrectly returned in queries after adding non-null columns in DDL operations #9084 @Lloyd-Pottiger
    - Fix the issue that the SUBSTRING_INDEX() function might cause TiFlash to crash in some corner cases #9116 @wshwsh12
    - Fix the issue that a large number of duplicate rows might be read in FastScan mode after importing data via BR or TiDB Lightning #9118 @JinheLin
- Tools
    - Backup & Restore (BR)
        - Fix the issue that BR fails to restore a transactional KV cluster due to an empty EndKey #52574 @3pointer
        - Fix the issue that a PD connection failure could cause the TiDB instance where the log backup advancer owner is located to panic #52597 @YuJuncen
        - Fix the issue that log backup might be paused after the advancer owner migration #53561 @RidRisR
        - Fix the issue that BR fails to correctly identify errors due to multiple nested retries during the restore process #54053 @RidRisR
        - Fix the issue that the connection used to fetch TiKV configurations might not be closed #52595 @RidRisR

        - Fix the issue that the TestStoreRemoved test case is unstable #52791 @YuJuncen
        - Fix the issue that TiFlash crashes during point-in-time recovery (PITR) #52628 @RidRisR
        - Fix the inefficiency issue in scanning DDL jobs during incremental backups #54139 @3pointer
        - Fix the issue that the backup performance during checkpoint backups is affected due to interruptions in seeking Region leaders #17168 @Leavrth
    - TiCDC
        - Fix inaccurate display of the **Kafka Outgoing Bytes** panel in Grafana #10777 @asddongmen
        - Fix the issue that data inconsistency might occur when restarting Changefeed repeatedly when performing a large number of UPDATE operations in a multi-node environment #11219 @lidezhu
    - TiDB Data Migration (DM)
        - Fix the connection blocking issue by upgrading go-mysql #11041 @D3Hunter
        - Fix the issue that SET statements cause DM to panic during the migration of MariaDB data #10206 @dveeden
    - TiDB Lightning
        - Fix the issue that TiDB Lightning might report an error when importing zstd compressed files #53587 @lance6716
    - Dumpling
        - Fix the issue that Dumpling reports an error when exporting tables and views at the same time #53682 @tangenta
    - TiDB Binlog
        - Fix the issue that deleting rows during the execution of ADD COLUMN might report an error data and columnID count not match when TiDB Binlog is enabled #53133 @tangenta

4.6 Contributors

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

- CabinfeverB
- DanRoscigno (First-time contributor)
- ei-sugimoto (First-time contributor)
- eltociear
- jiyfhust
- michaelmdeng (First-time contributor)
- mittalrishabh
- onlyacat
- qichengzx (First-time contributor)
- SeaRise
- shawn0915
- shunki-fujita (First-time contributor)
- tonyxuqqi
- wwu (First-time contributor)
- yzhan1