Oracle® NoSQL Database

Changelog

Release 22.2
E91819-27
August 2022

Oracle NoSQL Database Changelog, Release 22.2

E91819-27

Contents

Upgrade Requirements

Release 22.2 supports upgrades from releases for the prior two calendar years. To upgrade a store directly to the current release, the store must be running release 20.1 or later.

If you have a store running a 19.x release, we recommend that you upgrade it to the current release by first upgrading to the 21.2 or 20.3 release (or another 21.x or 20.x release) and then upgrading from that to the current release. If you have a store running an 18.x release, we recommend that you upgrade to the 20.3 release (or another 20.x release) and then upgrade from that to the current release. If you have a store running a 4.x release, we recommend that you first upgrade to the 19.5 release (or another 19.x release) and then follow our recommendations for upgrading from 19.x. If you have a store running a release earlier than the 4.x release and you are an Enterprise Edition user, please contact support. If you are a Community Edition user with this issue, please post a question to the NoSQL Database Discussions forum on Oracle Communities.

For more information, see the section on Upgrading an Existing Oracle NoSQL Database Deployment section in the Admin Guide.

Changes in 22.2.12

The following changes were made in Oracle NoSQL Database Release 22.2.12 Enterprise Edition.

Topics

New Features

1. The Rep Node parameter 'enableErasure' now defaults to true, meaning that Obsolete Data Erasure is now enabled by default.
   When 'enableErasure' is set to true, this enables the erasure feature for the underlying BDB JE storage layer. Erasure periodically wipes the obsolete data from the storage layer, by zeroing out corresponding records. Obsolete data is data that is no longer returned in queries or scans because it has been replaced or expired, but still exists in the database. Note that only table and key/value data is erased. Table metadata is not subject to erasure.

   The Rep Node parameter 'erasurePeriod' specifies the duration for one complete erasure pass over data. Erasure is throttled based on this value, to minimize its impact on performance. By default, it is set to "6 DAYS".

   [KVSTORE-1483]

Bug and Performance Fixes

1. Fixed a bug that in some cases could cause the XRegion Service to block for up to one hour before processing the next request to create or delete a table.
   [KVSTORE-1595]
2. Fixed a bug that could cause elasticity operations to fail in some cases where partitions were moved back and forth between two shards.
   [KVSTORE-1547]
3. Fixed a bug that a multi-region table would not be initialized correctly in some cases when the table was dropped and recreated in a remote region.
   [KVSTORE-1535]
4. Added 95% and 99% percentile latency values to the XRegion Service statistics.
   [KVSTORE-1404]
5. Fixed a bug in multi-region tables that, in some cases with high load, caused requests to create, alter and drop multi-region tables to become blocked inside the XRegion Service.
   [KVSTORE-1559]
6. Fixed a bug in multi-region tables that stream operations from remote regions could in some cases be lost when the XRegion Service writes them to the local region.
   [KVSTORE-1542]
7. Discovered that a bug fixed in the 21.1 release, but not previously reported, could have user-visible consequences. Prior to that release, checkpointing a multi-region table might cause some data to be lost during replication. In other words, some writes from one region might not be replicated to remote regions.

[KVSTORE-772]

8. Discovered that a bug fixed in the 21.1 release, but not previously reported, can have user-visible consequences. Prior to that release, checkpointing a multi-region table might cause a deadlock in data replication. If the issue is encountered, writes made to a multi-region table in one region may stop replication to remote regions, and network connections from remote regions might drop due to inactivity. *[KVSTORE-1525]*

9. Fixed a problem that could cause a thread deadlock in the direct Java driver for applications that perform queries. The problem was seen in the httpproxy, which uses the direct Java driver for queries, but is not specific to it. The result was that the application would hang.
   Using the `jstack` utility to create a thread dump for the application showed two characteristic stack traces, which represent the two sides of the deadlock:

```
"nioEventLoopGroup-3-11" #70 prio=10 os_prio=0 cpu=66.97ms
elapsed=5790.89s tid=0x00007f6fdc017800 nid=0x9e in Object.wait()
[0x00007f6fecbf3000]
   java.lang.Thread.State: RUNNABLE
    at
oracle.kv.impl.api.table.FieldDefImpl.<clinit>(FieldDefImpl.java:90)
    at
oracle.kv.impl.query.types.TypeManager.<clinit>(TypeManager.java:378
)
    at
oracle.kv.impl.query.compiler.FuncAndOr.<init>(FuncAndOr.java:38)
    at
oracle.kv.impl.query.compiler.FunctionLib.<init>(FunctionLib.java:16
9)
    at
oracle.kv.impl.query.compiler.CompilerAPI.<clinit>(CompilerAPI.java:
41)
    at
oracle.nosql.proxy.sc.TableUtils.getCallbackInfo(TableUtils.java:567
)
    at
oracle.nosql.proxy.DataService.handlePrepare(DataService.java:2032)
    at
oracle.nosql.proxy.DataService$$Lambda$354/0x0000000800551840.handl
e(Unknown Source)
    at
oracle.nosql.proxy.DataService.handleRequestWithContext(DataService.
java:664)
    at
oracle.nosql.proxy.DataService.handleRequestInternal(DataService.jav
a:451)
    at
oracle.nosql.proxy.DataService.handleRequest(DataService.java:424)
    at
oracle.nosql.proxy.DataService.handleRequest(DataService.java:399)
    at
oracle.nosql.proxy.DataService.handleRequest(DataService.java:355)
```

And:

```
"nioEventLoopGroup-3-9" #66 prio=10 os_prio=0 cpu=124.03ms
elapsed=5791.14s tid=0x00007f6fdc015800 nid=0x98 in Object.wait()
[0x00007f7163aef000]
   java.lang.Thread.State: RUNNABLE
    at
oracle.kv.impl.api.table.FieldDefSerialization.readTimestamp(FieldDefSeria
lization.java:433)
    at
oracle.kv.impl.api.table.FieldDefSerialization.readFieldDef(FieldDefSerial
ization.java:346)
    at
oracle.kv.impl.api.table.FieldDefSerialization.readFieldDef(FieldDefSerial
ization.java:261)
    at
oracle.kv.impl.api.table.FieldDefSerialization.readRecord(FieldDefSerializ
ation.java:380)
    at
oracle.kv.impl.api.table.FieldDefSerialization.readFieldDef(FieldDefSerial
ization.java:348)
    at
oracle.kv.impl.api.table.FieldDefSerialization.readFieldDef(FieldDefSerial
ization.java:261)
    at
oracle.kv.impl.query.runtime.PlanIter.deserializeFieldDef(PlanIter.java:11
84)
    at
oracle.kv.impl.query.runtime.ReceiveIter.<init>(ReceiveIter.java:367)
    at
oracle.kv.impl.query.runtime.PlanIter.deserializeIter(PlanIter.java:876)
    at
oracle.kv.impl.api.query.PreparedStatementImpl.<init>(PreparedStatementImp
l.java:550)
    at
oracle.nosql.proxy.DataServiceHandler.deserializePreparedQuery(DataService
Handler.java:810)
    at
oracle.nosql.proxy.DataService.deserializePreparedQuery(DataService.java:2
181)
    at oracle.nosql.proxy.DataService.handleQuery(DataService.java:1661)
    at
oracle.nosql.proxy.DataService$$Lambda$353/0x0000000800552440.handle(Unkno
wn Source)
    at
oracle.nosql.proxy.DataService.handleRequestWithContext(DataService.java:6
64)
    at
oracle.nosql.proxy.DataService.handleRequestInternal(DataService.java:451)
    at oracle.nosql.proxy.DataService.handleRequest(DataService.java:424)
```

[KVSTORE-1484]

10. Fixed a bug where using Ctrl+C or Ctrl+D to terminate paginated output of a query running in the SQL shell would cause a NullPointerException.

[KVSTORE-1521]

11. Fixed an issue that prevented a storage node from starting when using Java 8 after calling the securityconfig merge-trust utility to merge a security configuration with Java keystores in PKCS12 format into a configuration with Java keystores in JKS format. The start command would fail with following message:

    ```
    Failed to start SNA: Error contructing RMISocketPolicy using
    transport class for transport client
    ```

    [KVSTORE-1594]

12. Fixed a bug that the table description wrongly showed that JSON MRCounter fields were nullable. Users always need to set values for JSON MRCounters.
    [KVSTORE-1456]

13. Fixed a bug that caused creating a child table that included a JSON MRCounter to fail with the following error message:

    ```
    Error: Error found when creating the table: Only multi-region
    tables support MR_counters.
    ```

    [KVSTORE-1458]

14. Modified the implementation of `KVStore.executeSync` to reduce its thread usage by having it use async network operations.
    [KVSTORE-1072]

Utility Changes

1. Added the "show ddl <table>" command to the SQL shell. The new command returns the DDL of a table and its indexes, if any.
   [KVSTORE-1479]

2. The deprecated import/export utility and its related jar files (kvtool.jar, kvmigrator.jar, migrator.jar) have been removed and are no longer available. The old utility has been replaced by the standalone migrator utility, which is an independent download. It is available on the general Oracle NoSQL Database download page.
   [KVSTORE-1493]

3. Fixed a regression that was discovered affecting query processing for a small number of Hive query types. Prior to fixing this regression, each query's state was always reset when the job properties were configured. Although this addressed an issue with Big Data SQL query processing, because Hive and Big Data SQL employ different code paths, it has recently been observed that resetting the query state when the job properties are configured can result in incorrect results for some Hive queries. To address this issue, the state from the most recent query is now reset at the beginning of query processing for the next query. This addresses both the old issue with Big Data SQL query processing and this new issue with Hive.
   [KVSTORE-372]

4. Version.fromByteArray() will now throw IllegalArgumentException if the byte[] argument is invalid. It would previously throw FaultException.
   [KVSTORE-1599]

Changes in 22.1.22

The following changes were made in Oracle NoSQL Database Release 22.1.22 Enterprise Edition.

Topics

- Bug and Performance Fixes

Bug and Performance Fixes

1. Fixed an issue that if KVStore handles were repeatedly created under UnknownHostException or UnresolvedAddressException, sock type file descriptors would be leaked.
   [KVSTORE-1515]
2. Fixed a problem that could cause a thread deadlock in the direct Java driver for applications that perform queries. The problem was seen in the httpproxy, which uses the direct Java driver for queries, but is not specific to it. The result was that the application would hang.
   Using the `jstack` utility to create a thread dump for the application showed two characteristic stack traces, which represent the two sides of the deadlock:

```
"nioEventLoopGroup-3-11" #70 prio=10 os_prio=0 cpu=66.97ms
elapsed=5790.89s tid=0x00007f6fdc017800 nid=0x9e in Object.wait()
[0x00007f6fecbf3000]
   java.lang.Thread.State: RUNNABLE
    at
oracle.kv.impl.api.table.FieldDefImpl.<clinit>(FieldDefImpl.java:90)
    at
oracle.kv.impl.query.types.TypeManager.<clinit>(TypeManager.java:378)
    at oracle.kv.impl.query.compiler.FuncAndOr.<init>(FuncAndOr.java:38)
    at
oracle.kv.impl.query.compiler.FunctionLib.<init>(FunctionLib.java:169)
    at
oracle.kv.impl.query.compiler.CompilerAPI.<clinit>(CompilerAPI.java:41)
    at
oracle.nosql.proxy.sc.TableUtils.getCallbackInfo(TableUtils.java:567)
    at oracle.nosql.proxy.DataService.handlePrepare(DataService.java:2032)
    at
oracle.nosql.proxy.DataService$$Lambda$354/0x0000000800551840.handle(Unkno
wn Source)
    at
oracle.nosql.proxy.DataService.handleRequestWithContext(DataService.java:6
64)
    at
oracle.nosql.proxy.DataService.handleRequestInternal(DataService.java:451)
    at oracle.nosql.proxy.DataService.handleRequest(DataService.java:424)
    at oracle.nosql.proxy.DataService.handleRequest(DataService.java:399)
    at oracle.nosql.proxy.DataService.handleRequest(DataService.java:355)
```

And:

```
"nioEventLoopGroup-3-9" #66 prio=10 os_prio=0 cpu=124.03ms
elapsed=5791.14s tid=0x00007f6fdc015800 nid=0x98 in Object.wait()
[0x00007f7163aef000]
   java.lang.Thread.State: RUNNABLE
    at
oracle.kv.impl.api.table.FieldDefSerialization.readTimestamp(FieldDe
fSerialization.java:433)
    at
oracle.kv.impl.api.table.FieldDefSerialization.readFieldDef(FieldDef
Serialization.java:346)
    at
oracle.kv.impl.api.table.FieldDefSerialization.readFieldDef(FieldDef
Serialization.java:261)
    at
oracle.kv.impl.api.table.FieldDefSerialization.readRecord(FieldDefSe
rialization.java:380)
    at
oracle.kv.impl.api.table.FieldDefSerialization.readFieldDef(FieldDef
Serialization.java:348)
    at
oracle.kv.impl.api.table.FieldDefSerialization.readFieldDef(FieldDef
Serialization.java:261)
    at
oracle.kv.impl.query.runtime.PlanIter.deserializeFieldDef(PlanIter.j
ava:1184)
    at
oracle.kv.impl.query.runtime.ReceiveIter.<init>(ReceiveIter.java:367
)
    at
oracle.kv.impl.query.runtime.PlanIter.deserializeIter(PlanIter.java:
876)
    at
oracle.kv.impl.api.query.PreparedStatementImpl.<init>(PreparedStatem
entImpl.java:550)
    at
oracle.nosql.proxy.DataServiceHandler.deserializePreparedQuery(DataS
erviceHandler.java:810)
    at
oracle.nosql.proxy.DataService.deserializePreparedQuery(DataService.
java:2181)
    at
oracle.nosql.proxy.DataService.handleQuery(DataService.java:1661)
    at
oracle.nosql.proxy.DataService$$Lambda$353/0x0000000800552440.handl
e(Unknown Source)
    at
oracle.nosql.proxy.DataService.handleRequestWithContext(DataService.
java:664)
    at
oracle.nosql.proxy.DataService.handleRequestInternal(DataService.jav
a:451)
```

```
        at oracle.nosql.proxy.DataService.handleRequest(DataService.java:424)
```

[KVSTORE-1484]

3. Discovered that a bug fixed as of the initial 21.1 release can have user-visible consequences. The checkpointing of a multi-region table might cause some data to be lost during replication. In other words, some writes from one region might not be able to replicate to remote regions.
   [KVSTORE-772]
4. Discovered that a bug fixed as of the initial 21.1 release can have user-visible consequences. The checkpointing of a multi-region table might cause deadlock in data replication. If multi-region table runs into the bug, the writes made to one region may stop replication to remote regions, and network connections from remote regions might drop due to inactivity.
   [KVSTORE-1525]

Changes in 22.1.16

Topics

New Features

1. Added new classes that allow programmatic creation and control of a small, standalone store instance, comprising a single partition. The major classes are

   - KVLocal - represents the store instance
   - KVLocalConfig - allows configuration of the KVLocal instance

   KVLocal can be configured to use TCP/IP connections or, if using Java 16 or higher, Unix domain sockets.

   In support of KVLocal, the requirement that all keys specified in Operation and TableOperation lists passed to KVStore.execute and TableAPI.execute, respectively, share the same Major Path or shard key is relaxed for single partition stores such as the ones managed by KVLocal.

   [KVSTORE-1064]

2. Added three new SQL functions to do arithmetic operations on Timestamps:

   - `timestamp(9) timestamp_add(timestamp, string)`

     Adds a duration to a timestamp value and returns the new timestamp. The duration is a string with format [-]<n> (<UNIT>)+, where the <UNIT> can be YEAR, MONTH, DAY, HOUR, MINUTE, SECOND, MILLISECOND, NANOSECOND or the plural form of these keywords (e.g. YEARS). The UNIT keywords are case-insensitive. (e.g. 1 day 12 hours).

   - `long timestamp_diff(timestamp, timestamp)`

     Returns the number of milliseconds between two timestamp values (timestamp1 - timestamp2).

   - `string get_duration(long)`

     Converts the given number of milliseconds to a duration string.

   [KVSTORE-1338]

3. In SQL queries, allow casting of a timestamp value to a long value that is the number of milliseconds since January 1, 1970, 00:00:00 GMT.
   [KVSTORE-1247]

4. Added support for child tables to multi-region tables. Child tables always inherit the regions of the associated top level tables, so users cannot specify regions when creating child tables, nor can they explicitly add or drop regions for child tables.

[KVSTORE-1114]

5. Added support for MRCounter in JSON so that the MR_COUNTER data type can be used inside a schemaless JSON field. The syntax to declare an MR_COUNTER field for a JSON column is:

   ```
   field-name AS field-type MR_COUNTER
   ```

   A JSON MR_COUNTER can be updated the same way as any other MR_COUNTER column.

   [KVSTORE-1223]

6. Switched the KeyStore and TrustStore type of the default security configuration created by makebootconfig or securityconfig utility to PKCS12.
   The KeyStores and TrustStores used by the NoSQL Database server in the default security configuration will be created as PKCS12 password-protected stores. The TrustStore used by client applications (client.trust) in the default security configuration will be created as a PKCS12 password-less store if that feature is supported by the version of Java used to run the configuration utility. If not, the utilities will fall back to creating a JKS store if no password is specified for client.trust.

   Two new optional flags are introduced for security configuration creation:

   - `-kstype <JKS | PKCS12>`

     Create KeyStores and TrustStores in the security configuration as PKCS12 or JKS store, default PKCS12.

   - `-ctspwd <client.trust password>`

     Create client.trust as a password-protected PKCS12 store using the specified password.

   Upgrading to this release won't convert the store type of the existing KeyStores or TrustStores created by previous releases to PKCS12 automatically. To switch the store type of an existing configuration, a new command has been added to securityconfig utility:

   ```
   securityconfig config update -secdir security -kstype PKCS12
   ```

   Added two new client security properties to configure the password for client.trust if it's created as a password-protected PKCS12 store.

   - `oracle.kv.ssl.trustStorePassword`

   - `oracle.kv.ssl.trustStorePasswordAlias`

   The password of client.trust can be specified either directly via `oracle.kv.ssl.trustStorePassword` or kept in the password store. If the latter, specify the alias name of the password in the password store via the `oracle.kv.ssl.trustStorePasswordAlias` parameter, and the NoSQL Database Driver will retrieve the password from the password store automatically.

Updated the merge-trust command in securityconfig utility. When merging a security configuration created with a PKCS12 store into a security configuration created with a JKS store, this command will automatically convert the store type of the TrustStore to PKCS12 after merging all trusted certificates. However, this command won't convert the store type when merging trusted certificates of security configurations that are both created with JKS store.

[KVSTORE-60]

Bug and Performance Fixes

1. Fixed a bug that could have caused a NullPointerException when executing "describe table" against a multi-region table.
   [KVSTORE-1426]
2. Fixed a bug where adding or removing regions from a multi-region table might not have been executed correctly when the store was under high stress.
   [KVSTORE-1374]
3. Fixed a bug where if a key-only table was later altered to add an identity column, an error would be raised if an attempt was made to read a key-only row that was inserted before the alter.
   [KVSTORE-1370]
4. Fixed a bug where implicit casting of a string to a timestamp does not work when the string comes from JSON data.
   [KVSTORE-1477]
5. Fixed a bug that the default value of an internal JE parameter preHeartbeatTimeoutMs is not set to a proper value. This bug will cause rare cases that the store takes more than one hour to recover from faults or a master transfer.
   [KVSTORE-1384]
6. Fixed a bug that when a key-only MR table evolves by adding new columns, the newly added columns might be lost when being replicated to remote regions.
   [KVSTORE-1333]
7. Fixed a problem where a row with a homogeneous JSON array of type STRING written with release 20.2 or earlier releases would fail to deserialize in release 20.3 or later releases. The issue shows up as a row that is not found in the result of a get() or in results of queries that need to deserialize the row value. The data remains intact but is not accessible.
   [KVSTORE-1347]
8. Fixed a bug in SQL INSERT when the table has an auto-generated identity or UUID column and a column list is used before the VALUES clause. In this case, an exception would be thrown, but should not have been.
   [KVSTORE-1371]
9. Fixed a query bug when the partition() function is used in the WHERE clause. The following example demonstrates the bug:

   ```
   select id, partition($f) as part from foo $f
   where $f.address.state = "MA" and partition($f) = 2
   ```

   The above query will be sent for execution to an RN that contains partition 2. If there is an index on address.state, the query will use that index at that RN. In general, the index will contain entries from multiple partitions that are stored at the

RN. The bug was that while the index was scanned, entries that do not belong to partition 2 were not filtered out.

[KVSTORE-1334]

10. Fixed a bug that prevented a table with a UUID column from being altered after the table was schema evolved: rows inserted before the schema evolution would not deserialize after the schema evolution.
[KVSTORE-1357]

11. Fixed a bug in computing the result type of a query. Specifically, the nullability flag of a field in the result type could be wrongly set to false. This would happen if the SELECT-clause expression that corresponds to that field may not return a NULL, but may return an empty result. The nullability flag should be set to true in this case because the SELECT clause converts EMPTY to NULL.
[KVSTORE-1342]

12. Fixed a bug where plans in the APPROVED state would not be run after an Admin restart.
[KVSTORE-1194]

13. Fixed a bug where the Streams API could lose writes to a new shard created during a store expansion.
[KVSTORE-1336]

14. Fixed a problem that could occur if the XRegion Agent was creating a new multi-region table and, at the same time, it found a previously missing multi-region table in a remote region. In this case, the agent could fail to add the remote region to the table, and the local table would not receive updates from the remote region.
[KVSTORE-1300]

15. Fixed compatibility problems in deserializing the server-side execution plan of proxy-based queries. Problems involving two cases are fixed:

    a. A query is prepared with a kvclient that is newer than the servers.

    b. A query is prepared with a kvclient at version V1. The prepared query is cached by the application. Then, kvclient and kvserver are upgraded to V2. The application executes the cached query again. In this case, the servers would attempt (and fail) to deserialize the query plan using V2, instead of V1.

    [KVSTORE-1363]

16. Fixed a bug where adding a multi-region table could fail when there were network or remote region failures. The XRegion agent now has a retry mechanism so that it will add the table when the failures are resolved.
Also fixed a bug that when a Streams API subscription reconnects to a store, the subscription may resume from a position later than expected. This problem could affect both the Streams API and multi-region tables.

    [KVSTORE-1265]

17. Fixed an incompatible change introduced in 20.3 in the performance JSON files (e.g., rnOp*.json). The incompatible change was that latency metrics (for example AllSingleKeyOperations_Interval_Avg) were changed to integer values where they were previously floating-point numbers. This change caused sub-millisecond latency values to appear as zero. Fixed by reverting back to using floating-point numbers.
[KVSTORE-1289]

18. Fixed a problem where a query run on a table with data that was created prior to release 19.5 and never modified might fail with an error that indicates "modification time not available".

[KVSTORE-1234]

19. Fixed a bug with unnesting queries. Query results would be lost if the query was not using an unnesting index and the batch size was reached in the midst of unnesting the arrays or maps specified in the query.
[KVSTORE-1228]

20. Fixed a bug with unnesting queries. Queries that specified `count(*)` were incorrectly using an unnesting index that unnested at a deeper level than the query.
[KVSTORE-1292]

21. Fixed two bugs related to query timeouts. The first problem was that a query could timeout at the driver but keep running at the RNs. The second problem was that the query timeout would not be respected if the query performed a generic (non-index-based) sort or group by.
[KVSTORE-353]

22. Modified the output of the 'show admins' Admin CLI command to display the admin storage directory path, and of the 'ping' command to display the size of the admin storage directory.
[KVSTORE-102]

23. Added TLSv1.3 to the default TLS protocols of security configurations created with the makebootconfig or securityconfig utilities. The default TLS protocols for security configuration are now "TLSv1.3, TLSv1.2". Note that the new protocols are only used when creating new security configurations with the above utilities. Upgrading to this release will not automatically update the existing security configuration of a running store. We recommended upgrading existing stores to use the TLSv1.3 protocol: see "Guidelines for Enabling TLSv1.3 Protocol" in Security Guide of Oracle NoSQL Database.
[KVSTORE-263]

24. The makebootconfig and securityconfig utilities now generate 2048 byte RSA keys. Note that updated keys are only created when generating new security configurations using the above utilities. Upgrading to this release will not automatically update the existing private keys used by a running store. We recommended upgrading existing stores to use new private keys so that they have 2048 byte RSA keys: see "Guidelines for Updating SSL Keys and Certificates" in Security Guide of Oracle NoSQL Database.
[KVSTORE-932]

25. Modified the output from the "describe table" DDL command to include multi-region table information if the table is multi-region.
[KVSTORE-1155]

26. Made changes to prevent a multi-region table that is missing at a remote region from disrupting existing multi-region tables in local and other remote regions. If an MR table is missing or has a incompatible schema in a remote region, with the fix the agent will check the particular table periodically while, at the same time, continuing to service other remote regions where the MR table is available. When the table in question is available again in the remote region, for example if the missing table is recreated or the incompatible schema is fixed, the agent will include the table and service it like other available MR tables.
[KVSTORE-395]

27. Fixed a bug that prevented key-only multi-region tables from working properly.
[KVSTORE-1302]

28. Fixed several bugs with partition migration, which is the underlying mechanism to move user data from one shard to another during store elasticity operations. The following problems were fixed:

    - Record modification times could be migrated incorrectly, which caused problems with multi-region tables
    - Client writes could be lost at the end of the migration under some edge cases
    - The migration service might not be available after certain RN replication state changes.

    [KVSTORE-1250] [KVSTORE-1244] [KVSTORE-1221]

29. Improved the robustness of the Java direct driver in handling network connection problems it encounters when contacting storage nodes.
    [KVSTORE-1423]

30. Fix a bug that could cause RepNode crashes or RepNode metadata corruption in a short time window after an upgrade during which the store version had not been fully updated. This bug could cause RepNodes to crash if a new table with UUIDs or MRCounters was added, or an existing table was altered to add UUID or MRCounter fields, during the time window. For tables with UUIDs, RepNodes would crash during the time window, but the problem would fix itself after the store version was updated. For tables with MRCounters, the bug could cause persistent metadata corruption, which would result in RepNodes continuing to crash on restart.
    [KVSTORE-1361]

API Changes

1. `Row.getExpirationTime()` no longer throws an exception if the underlying Row metadata is not available. Instead it will return 0 indicating no expiration time. Expiration time is 0 for rows that either do not have a TTL or were written prior to the introduction of the TTL feature.
   Also added `Row.getLastModificationTime()` to return the last modification time of a Row in milliseconds since January 1, 1970. If the row was last modified using a version of the system older than release 19.5, the modification time is not available and 0 is returned.

   [KVSTORE-1234]

2. Marked the `IndexKeySizeLimitException` class as hidden; it is for internal use only.
   [NOSQL-408]

Changes in 21.2.46

The following changes were made in Oracle NoSQL Database Release 21.2.46 Enterprise Edition.

Topics

Bug and Performance Fixes

1. Fixed a bug preventing tables with UUID columns from being altered: rows inserted before the schema evolution would not deserialize after the schema evolution.
   [KVSTORE-1357]
2. Fixed a bug where if a key-only table was later altered to add an identity column, an error would be raised if an attempt was made to read a key-only row that was inserted before the alter.
   [KVSTORE-1370]
3. Fixed a bug in SQL INSERT when the table has an auto-generated identity or UUID column and a column list is used before the VALUES clause. In this case, an exception would be thrown, erroneously.
   [KVSTORE-1371]
4. Fixed compatibility problems in deserializing the server-side execution plan of proxy-based queries. Two problems are fixed:
   a. A query is prepared with a kvclient that is newer than the servers.
   b. A query is prepared with a kvclient at version V1. The prepared query is cached by the application. Then, kvclient and kvserver are upgraded to V2. The apllication executes the cached query again. In this case, the servers would attempt (and fail) to deserialize the query plan using V2, instead of V1.

   [KVSTORE-1363]
5. Fixed an issue that if KVStore handles were repeatedly created under UnknownHostException or UnresolvedAddressException, sock type file descriptors would be leaked.
   [KVSTORE-1515]

Changes in 21.2.29

Topics

Bug and Performance Fixes

1. Fixed a problem where a row with an homogenous JSON array of type STRING written with 20.2 or earlier would fail to deserialize in 20.3 or later. The issue shows up a row that is not found in get() or queries that need to deserialize the row value. The data remains intact but is not accessible.
   [KVSTORE-1347]
2. Fixed a problem where a query run on a table with data that was created prior to release 19.5 and never modified might fail with an error that indicates "modification time not available." The same error might occur if the Row.getLastModificationTime() call is made on such a row.
   [KVSTORE-1234]

API Changes

1. Row.getExpirationTime() and Row.getLastModificationTime() no longer throw an exception if the underlying Row metadata is not available. Instead they will return 0 indicating no expiration and no available modification time, respectively. Expiration time is 0 for rows that either do not have a TTL or were written prior to the introduction of the TTL feature. Modification time of 0 occurs only for rows that were written prior to the introduction of modification time, which was release 19.5.
   [KVSTORE-1234]
2. Marked IndexKeySizeLimitException as hidden, it is for internal use only.
   [NOSQL-408]

Changes in 21.2.19

Topics

New Features

1. An off-heap cache is no longer used by Oracle NoSQL Database. The off-heap cache did not provide the performance benefits we had expected, and in some cases had a negative performance impact, so it has been removed.
   As part of this change, the `systemPercent` parameter, which was previously used to configure the size of the off-heap cache, has been made obsolete. The parameter may still be specified, but it will be ignored.

   For stores with configurations where memory was allocated to the off-heap cache, that memory is now available for file system caching in the system buffer pool. In other words, data formerly stored in the off-heap cache will now be cached in the system buffer pool.

   [KVSTORE-927]

2. Implemented unnesting of arrays and maps.
   The FROM clause has been extended to allow a list of expressions after the table or tables referenced by the query. Conceptually, for each table row, the expressions are evaluated, and a cross product is formed by the current row and the results of these evaluations. The result of the whole FROM clause is a table containing the union of all these cross products.

   Although any kind of expression may appear in the FROM clause, the primary purpose of this feature is to unnest arrays (or maps) that appear in table rows. The main reason to support nesting is to group by fields that are contained in the arrays or maps. As an example, consider the following table that stores data from a TV streaming service:

   ```
   create table stream_acct(acct_id integer, value json, primary
   key(acct_id))
   ```

   A sample row of this table looks like this:

   ```
   {
     "acct_id": 1,
     "value": {
       "firstName" : "John",
       "lastName" : "Sanders",
       "country" : "USA",
       "contentStreamed": [
         {
           "showName": "Call My Agent",
           "showId": 15,
           "showType": "tvseries",
           "numSeasons" : 2,
   ```

```
        "seriesInfo": [
          {
            "seasonNum" : 1,
            "numEpisodes" : 2,
            "episodes": [
              { "episodeID" : 20, "lengthMin" : 45, "minWatched" : 45 },
              { "episodeID" : 30, "lengthMin" : 42, "minWatched" : 42 }
            ]
          },
          {
            "seasonNum": 2,
            "numEpisodes" : 2,
            "episodes": [
              { "episodeID" : 20, "lengthMin" : 50, "minWatched" : 50 },
              { "episodeID" : 30, "lengthMin" : 46, "minWatched" : 46 }
            ]
          }
        ]
      },
      {
        "showName": "Rita",
        "showId": 16,
        "showType": "tvseries",
        "numSeasons" : 1,
        "seriesInfo": [
          {
            "seasonNum" : 1,
            "numEpisodes" : 2,
            "episodes": [
              { "episodeID" : 20, "lengthMin" : 65, "minWatched" : 65 },
              { "episodeID" : 30, "lengthMin" : 60, "minWatched" : 60 }
]
          }
        ]
      }
    ]
  }
}
```

The following query orders the shows in USA according to their "popularity", i.e., how many users have shown an interest in them.

```
select $show.showId, count(*) as cnt
from stream_acct $s, unnest($s.value.contentStreamed[] as $show)
where $s.value.country = "USA"
group by $show.showId
order by count(*) desc
```

The above query can be optimized if the following index is created:

```
create index idx_country_showid on stream_acct(
    value.country as string,
    value.contentStreamed[].showId as integer) with unique keys per row
```

The query will use this index. The country condition will be pushed to the index, the group-by will be index-based, and the index is a covering one for this query. Notice that, for the index to be used, the index keys extracted per row must not contain any duplicates. This constraint is enforced by creating the index as "with unique keys per row".

Notice that the unnest clause in the above query (and in general) is not strictly necessary; it can be omitted without changing the semantics (and the results) of the query. However, if it is omitted, the index will not used. This is because the unnest clause puts some restrictions on the kind of expressions it wraps, and these restrictions make it possible for the query processor to find that an index on the arrays/maps that are being unnested is applicable and rewrite the query accordingly.

[KVSTORE-822]

3. Added `seq_distinct()` function. It returns the distinct values of the sequence produced by its input expression.
   As an example, the following query returns, for each area code, the number of users having phone numbers in that area code. The query uses the seq_distinct() because a user may have multiple phone numbers in the same area code, but he or she must be counted only once per area code.

   ```
   select $areacode, count(*) as cnt
   from Users u, seq_distinct(u.address.phones[].areacode) as $areacode
   group by $areacode
   ```

   [KVSTORE-822]

4. In comparison operators, when one operand is a timestamp and the other a string, the string operand is now implicitly cast to a timestamp with the same precision as the precision of the timestamp operand. An exception is thrown if the cast fails. *[KVSTORE-1016]*

5. Enhanced support for nested seq_transform() expressions.
   `seq_transform()` is useful for transforming JSON documents stored in table rows. In such situations it is often the case that multiple `seq_transform()` expressions must be used, nested into each other. Furthermore, inner `seq_transform()` expressions may need to access the "current" input item of outer `seq_transform()` expressions. Until now this was not possible. In this release we enhance `seq_transform()`, so that each such expression creates an appropriately named context-item variable that can be referenced by inner `seq_transform()` expressions.

   For example, consider the following table that records information about air travel luggage:

   ```
   create table luggage (
     ticketNo string,
     passengerName string,
     bagInfo json,
     primary key(ticketNo))
   ```

A sample row for this table is shown below


```
{
  "ticketNo" : "1762352483606",
  "passengerName" : "Willie Hernandez",
  "bagInfo" : [
    {
      "tagNum" : "17657806243915",
      "routing" : "SFO/AMS/HER",
      "lastActionCode" : "offload",
      "lastSeenStation" : "HER",
      "lastSeenTimeGmt" : "2019-03-13T15:19:00",
      "flightLegs" : [
        {
          "flightNo" : "BM604",
          "flightDate" : "2019-03-12T20:00:00",
          "fltRouteSrc" : "SFO",
          "fltRouteDest" : "AMS",
          "estimatedArrival" : "2019-03-13T08:00:00",
          "actions" : [
            { "at":"SFO", "action":"TagScan",
"time":"2019-03-12T18:14:00" },
            { "at":"SFO", "action":"onload",
"time":"2019-03-12T19:20:00" },
            { "at":"AMS", "action":"offload",
"time":"2019-03-13T08:30:00" }
          ]
        },
        {
          "flightNo" : "BM667",
          "flightDate" : "2019-03-13T11:14:00",
          "fltRouteSrc" : "AMS",
          "fltRouteDest" : "HER",
          "estimatedArrival" : "2019-03-13T15:00:00",
          "actions" : [
            { "at":"AMS", "action":"TagScan",
"time":"2019-03-13T10:45:00" },
            { "at":"AMS", "action":"onload",
"time":"2019-03-13T10:50:00" },
            { "at":"HER", "action":"offload",
"time":"2019-03-13T15:19:00" }
          ]
        }
      ]
    },
    {
      "tagNum" : "17657806244523",
      "routing" : "SFO/AMS/HER",
      "lastActionCode" : "offload",
      "lastSeenStation" : "AMS",
      "lastSeenTimeGmt" : "2019-03-13T08:35:00",
      "flightLegs" : [
        {
          "flightNo" : "BM604",
          "flightDate" : "2019-03-12T20:00:00",
```

```
          "fltRouteSrc" : "SFO",
          "fltRouteDest" : "AMS",
          "estimatedArrival" : "2019-03-13T08:00:00",
          "actions" : [
            { "at":"SFO", "action":"TagScan",
"time":"2019-03-12T18:14:00" },
            { "at":"SFO", "action":"onload",
"time":"2019-03-12T19:22:00" },
            { "at":"AMS", "action":"offload",
"time":"2019-03-13T08:32:00"  }
          ]
        }
      ]
    }
  ]
}
```

Let's say we want to write a query that will return, for each ticketNo, a flat array containing all the actions performed on the luggage of that ticketNo. However, we want only the "at" and "action" fields of each action. Furthermore, we want the flightNo and the tagNum fields to be included with each action. Specifically, for the above sample row, the result should be the following:

```
{
  "actions" : [
    {"at":"SFO", "action":"TagScan", "flightNo":"BM604",
"tagNum":17657806243915},
    {"at":"SFO", "action":"onload",  "flightNo":"BM604",
"tagNum":17657806243915},
    {"at":"AMS", "action":"offload", "flightNo":"BM604",
"tagNum":17657806243915},
    {"at":"AMS", "action":"TagScan", "flightNo":"BM667",
"tagNum":17657806243915},
    {"at":"AMS", "action":"onload",  "flightNo":"BM667",
"tagNum":17657806243915},
    {"at":"HER", "action":"offload", "flightNo":"BM667",
"tagNum":17657806243915},
    {"at":"SFO", "action":"TagScan", "flightNo":"BM604",
"tagNum":17657806244523},
    {"at":"SFO", "action":"onload",  "flightNo":"BM604",
"tagNum":17657806244523},
    {"at":"AMS", "action":"offload", "flightNo":"BM604",
"tagNum":17657806244523},
  ]
}
```

Here is the query that will return the above result:

```
select
  seq_transform(
      l.bagInfo[],
      seq_transform(
          $sql.flightLegs[],
          seq_transform(
```

```
                $sq2.actions[],
                {
                  "at" : $sq3.at,
                  "action" : $sq3.action,
                  "flightNo" : $sq2.flightNo,
                  "tagNum" : $sq1.tagNum
                }
            )
        )
    ) as actions
from luggage l
```

In this query, `$sq1` iterates over the elements of the bagInfo array, `$sq2` iterates over the elements of the flightLegs array, and `$sq3` iterates over the elements of the actions array.

[KVSTORE-1044]

Bug and Performance Fixes

1. Fixed an issue with KVStats that the `NodeMetrics.getAverageLatencyNanos` method always returns 10000.
   [KVSTORE-996]
2. Disabled JVM Biased Locking for RNs. Now -XX:-UseBiasedLocking will be explicitly passed to RNs' JVM cmdline. This change should make RNs give more predictable latencies. There should not be any performance degradation due to this change.
   [KVSTORE-1047]
3. The default partition migration concurrency has been increased. Partition migration is done during some topology changes, such as redistribute and rebalance. There are two parameters which limit how may partition migrations can take place concurrently. The default values for these parameters have been increased. Specifically:

   - `rnPMConcurrentSourceLimit`- default changed from 1 to 2
   - `rnPMConcurrentTargetLimit` - default changed from 2 to 4

   The higher level of concurrency, in most cases, will reduce the overall time that a topology change takes to complete. The increase should not negatively impact ongoing data operations during a topology change, especially when using SSDs. If the store is running on older hardware with rotating disks, the user may want to explicitly set the parameters to the original default values.

   [KVSTORE-1002]
4. The output of the Admin CLI commands `ping` and `verify configuration` now includes information about service start times and state change times.
   [KVSTORE-876]
5. The use of Java RMI in the direct Java driver is now optional. The driver now only uses RMI when accessing servers running software prior to the 21.2 release, or if the driver has been configured to disable use of the async network protocol by calling `KVStoreConfig.setUseAsync(false)`. Because the async network protocol supports concurrent calls on the same socket connection, this change should reduce socket and other resource usage in some cases, particularly for stores with large numbers of shards.
   [KVSTORE-340]
6. The `haHostname` and `haPortRange` Storage Node parameters are now read-only. Changing these parameters already had no effect; the new behavior is just intended to

make it clear that the parameters should not be changed. The only way to switch to new JE HA parameters is to migrate to a new Storage Node.
[KVSTORE-437]

7. There is a new flag to the kvlite command that allows users to control the storage size associated with a new kvlite store: `-storagedirsizegb size_in_GB`.
The default value is 10GB, which has changed from the previous 1GB default. This flag only applies for the initial creation of the store and is ignored when kvlite is started on an existing store.

   In addition, the output of kvlite when started on an existing store now properly reflects the values in the existing store.

   [KVSTORE-1042]

8. The `StringDef.equals()` method now considers subtypes when determining if two fields are equal. The types `"STRING"`, `"STRING AS UUID"` and `"STRING AS UUID GENERATED BY DEFAULT"` are now not equal.
[KVSTORE-1048]

9. Modified the `verify configuration` Admin CLI command to report version differences as notes, not violations. Version differences are expected during upgrades, so they should not be considered violations.
Modified the non-JSON output format of the `verify configuration` and `ping` Admin CLI commands to add spaces after the colon separators that appear at the end of keywords in the output, to improve readability. Also removed spaces from a couple keywords, changing "available storage size" to "availableStorageSize" and "storage type" to "storageType".

   Made changes to the non-JSON output format of Admin CLI commands related to login issues to make the output more consistent.

   [KVSTORE-448]

10. Modified the Admin CLI `plan deploy-topology` command to require that the new topology being deployed reduce the number of outstanding topology violations, if any, rather than just not increase the number. As before, the command can still be forced to deploy a new topology by specifying the -force flag. Also fixed a problem with the way the command counted topology violations that was inaccurate in some cases.
These two changes together fix a problem where deploying a new zone without providing sufficient SNs, or providing ones with insufficiently large storage directory sizes, could result in a topology deployment appearing to succeed without actually making the changes needed to populate the new zone.

    [KVSTORE-542]

11. Removed TLSv1 and TLSv1.1 from the default TLS protocol settings. New security configurations will be created with only TLSv1.2 by default starting with this release.
[KVSTORE-903]

12. Added two new Rep Node parameters 'enableErasure' and 'erasurePeriod' to control data erasure.
If 'enableErasure' is set to true, then this enables the erasure feature for the underlying BDB JE storage layer. Erasure periodically wipes the obsolete data from the storage layer, by zeroing out corresponding records. The default value is false. Note that only table and key/value data is erased. Table metadata is not subject to erasure.

When erasure is enabled, the value specified for the 'erasurePeriod' parameter specifies the duration for one complete erasure pass over data. Erasure is throttled based on this value, to minimize its impact on performance. By default, it is set to "6 DAYS".

[KVSTORE-1033]

13. Fixed an issue that causes some BigDataSQL queries to encounter a NullPointerException in certain cases. That issue was characterized by a stacktrace like the following:

```
Caused by: java.lang.NullPointerException
at java.util.Objects.requireNonNull(Objects.java:203)
at java.util.AbstractCollection.retainAll(AbstractCollection.java:405)
at
oracle.kv.impl.query.runtime.RuntimeControlBlock.setPartitionSet(RuntimeCo
ntrolBlock.java:258)
at oracle.kv.impl.query.runtime.ReceiveIter.open(ReceiveIter.java:1252)
at
oracle.kv.impl.api.query.QueryStatementResultImpl$QueryResultIterator.<ini
t>(QueryStatementResultImpl.java:297)
at
oracle.kv.impl.api.query.QueryStatementResultImpl.<init>(QueryStatementRes
ultImpl.java:153)
at
oracle.kv.impl.api.query.QueryStatementResultImpl.<init>(QueryStatementRes
ultImpl.java:115)
at
oracle.kv.impl.api.query.PreparedStatementImpl.executeSyncPartitions(Prepa
redStatementImpl.java:365)
at
oracle.kv.impl.api.KVStoreImpl.executeSyncPartitions(KVStoreImpl.java:3635
)
at
oracle.kv.hadoop.table.TableRecordReaderBase.getNextOnqlIteratorPartition
s(TableRecordReaderBase.java:523)
at
oracle.kv.hadoop.table.TableRecordReaderBase.getNextIterator(TableRecordRe
aderBase.java:500)
at
oracle.kv.hadoop.table.TableRecordReaderBase.nextKeyValue(TableRecordReade
rBase.java:391)
... 11 more
```

With this release, the issue above has been fixed and no longer occurs.

[KVSTORE-1123]

14. Fixed a issue that when there are multiple subscriptions for the Streams API on a shard, changes to the tables on a subscription, like adding or removing tables, would cause the same changes of tables to other subscriptions as well.
[KVSTORE-35]

15. Fixed a deadlock issue related to SSLDataChannel. The issue will produce stack traces similar to the following in a thread dump:

```
"ReplicaOutputThread" #829 daemon prio=5 os_prio=0 cpu=8695.34ms
elapsed=66577.11s tid=0x000055f030c1f800 nid=0x49f2 waiting for monitor
```

```
entry  [0x00007fc720171000]
   java.lang.Thread.State: BLOCKED (on object monitor)
    at
com.sleepycat.je.rep.utilint.net.SSLDataChannel$ChannelWriteTask.ru
n(SSLDataChannel.java:1425)
    - waiting to lock <0x00000010046df7e0> (a
java.nio.HeapByteBuffer)
    at
com.sleepycat.je.rep.utilint.net.SSLDataChannel$ChannelWriteTask.acc
ess$1100(SSLDataChannel.java:1413)
    at
com.sleepycat.je.rep.utilint.net.SSLDataChannel.flush(SSLDataChannel
.java:590)
    ......
"REPLICA rg15-rn2(2)" #68 daemon prio=5 os_prio=0 cpu=9106.65ms
elapsed=68622.76s tid=0x000055f030c55800 nid=0xb24 in
Object.wait()  [0x00007fc722b93000]
   java.lang.Thread.State: WAITING (on object monitor)
    at java.lang.Object.wait(java.base@11.0.8/Native Method)
    - waiting on <no object reference available>
    at java.lang.Object.wait(java.base@11.0.8/Object.java:328)
    at
com.sleepycat.je.rep.utilint.net.SSLDataChannel$ChannelTask.transitT
oInProgress(SSLDataChannel.java:1335)
    - waiting to re-lock in wait() <0x00000010046e01b8> (a
java.lang.Object)
    at
com.sleepycat.je.rep.utilint.net.SSLDataChannel$ChannelWriteTask.ru
n(SSLDataChannel.java:1423)
    at
com.sleepycat.je.rep.utilint.net.SSLDataChannel$ChannelWriteTask.acc
ess$1100(SSLDataChannel.java:1413)
    at
com.sleepycat.je.rep.utilint.net.SSLDataChannel.doWrap(SSLDataChanne
l.java:910)
    - locked <0x00000010046df7e0> (a java.nio.HeapByteBuffer)
    at
com.sleepycat.je.rep.utilint.net.SSLDataChannel.handshake(SSLDataCha
nnel.java:847)
    at
com.sleepycat.je.rep.utilint.net.SSLDataChannel.unwrap(SSLDataChanne
l.java:771)
    at
com.sleepycat.je.rep.utilint.net.SSLDataChannel.read(SSLDataChannel.
java:403)
    ......
```

[KVSTORE-1021]

16. Fixed a livelock issue related to SSLDataChannel. The issue is observed as an inactive channel with either SSLDataChannel doWrap or doUnwrap. Stack traces similar to the following will be seen in a thread dump:

```
   java.lang.Thread.State: RUNNABLE
        at java.lang.Object.hashCode(java.base@11.0.6/Native Method)
```

```
        at
java.util.concurrent.ConcurrentHashMap.replaceNode(java.base@11.0.6/
ConcurrentHashMap.java:1111)
        at java.util.concurrent.ConcurrentHashMap.remove(java.base@11.0.6/
ConcurrentHashMap.java:1102)
        at
com.sleepycat.je.utilint.LoggerUtils.logMsg(LoggerUtils.java:538)
        at
com.sleepycat.je.utilint.LoggerUtils.logMsg(LoggerUtils.java:445)
        at
com.sleepycat.je.rep.utilint.net.DataChannelFactoryBuilder$ChannelInstance
Logger.log(DataChannelFactoryBuilder.java:417)
        at
com.sleepycat.je.rep.utilint.net.SSLDataChannel$ChannelWriteTask.run(SSLDa
taChannel.java:1636)
        - locked <0x00000000f037cc80> (a java.lang.Object)
        at
com.sleepycat.je.rep.utilint.net.SSLDataChannel$ChannelWriteTask.access$14
00(SSLDataChannel.java:1613)
        at
com.sleepycat.je.rep.utilint.net.SSLDataChannel.doWrap(SSLDataChannel.java
:990)
    ......
   java.lang.Thread.State: RUNNABLE
        at java.lang.Object.notifyAll(java.base@11.0.6/Native Method)
        at
com.sleepycat.je.rep.utilint.net.SSLDataChannel$ChannelTask.transitToIdle(
SSLDataChannel.java:1513)
        - locked <0x00000000f039e788> (a java.lang.Object)
        at
com.sleepycat.je.rep.utilint.net.SSLDataChannel$ChannelReadTask.run(SSLDat
aChannel.java:1808)
        at
com.sleepycat.je.rep.utilint.net.SSLDataChannel$ChannelReadTask.access$180
0(SSLDataChannel.java:1751)
        at
com.sleepycat.je.rep.utilint.net.SSLDataChannel.readChannelForBufUnderflo
w(SSLDataChannel.java:1168)
        at
com.sleepycat.je.rep.utilint.net.SSLDataChannel.doUnwrap(SSLDataChannel.ja
va:1079)
        - locked <0x00000000f0386f20> (a java.lang.Object)
        - locked <0x00000000f0386ea8> (a java.lang.Object)
        at
com.sleepycat.je.rep.utilint.net.SSLDataChannel.handshake(SSLDataChannel.j
ava:930)
        at
com.sleepycat.je.rep.utilint.net.SSLDataChannel.unwrap(SSLDataChannel.java
:828)
        at
com.sleepycat.je.rep.utilint.net.SSLDataChannel.read(SSLDataChannel.java:4
41)
    ......
```

[KVSTORE-1022]

17. Fixed a problem where RepNode or MasterRebalanceThread in StorageNode may exit unexpectedly due to uncaught InternalFaultException(s) when transient network issues occur. In this case, the logging output shows both of them exit because of SessionAccessException, which is incorrect and has also been fixed.

    ```
    2021-02-12 10:16:26.575 UTC SEVERE [sn6] MasterRebalanceThread
    thread exiting due to exception.
    oracle.kv.impl.security.SessionAccessException:
    java.rmi.ConnectIOException: Exception creating connection to:
    ...

    2021-02-12 10:16:26.575 UTC SEVERE [sn6] MasterRebalanceThread
    thread exiting due to exception.
    oracle.kv.impl.security.SessionAccessException:
    java.rmi.ConnectIOException: Exception creating connection to:
    ...
    ```

    [KVSTORE-942]

18. Fixed a bug with the `DROP NAMESPACE` command where any existing tables in the namespace will also be dropped but their data remains in the store. Since the tables have been dropped their data can not be accessed or removed.
    [KVSTORE-1076]

19. Fixed a bug that using UPDATE statement with REMOVE clause to remove an element from ARRAY(RECORD) or MAP(RECORD) fails with java.lang.IllegalArgumentException: Error: at (...) Cannot remove fields from records.
    [KVSTORE-1200]

Packaging Changes

1. The packaging of kvclient.jar and kvstore.jar has been changed to include classes from Oracle internal libraries that are not available in public Maven repositories and were previously included as separate JAR files in the distribution. This change is intended to make it easier to create and use a Maven pom.xml file that depends on kvclient.jar since the JAR file now only has dependencies that are available publicly. The libraries included are:

    - commonutil.jar
    - sklogger.jar
    - sdoapi.jar
    - sdoutl.jar
    - sdodep3prt.jar

    In addition, classes from Enterprise Edition-only libraries referenced by kvstore-ee.jar are now directly included in that library, again for simplicity in handling dependencies. The libraries included are the ones associated with the Oracle Wallet:

    - osdt_core.jar
    - osdt_cert.jar
    - osraclepki.jar

Also, the XML build files have been removed from the Community Edition release as it is not currently possible to fully build from source.

[KVSTORE-1100]

Changes in 21.1.12

Topics

New Features

1. Added support for indexing nested arrays and map, and using such indexes in queries.

   Previously, Oracle NoSQL could index only one array or map per table row, meaning that an indexed array or map could not be nested inside another array or map. This release adds support for indexing nested arrays and maps. For example, consider the following table that records information about air travel baggage:

   ```
   create table baggage (
     ticketNo string,
     passengerName string,
     bagInfo json,
     primary key(ticketNo))
   ```

   A sample row for this table is shown below. In this row, there is only one piece of baggage for ticketNo "1762352483606", but in general, passengers may have more than one piece of baggage, and as a result, "bagInfo" is an array of documents.

   ```
   {
     "ticketNo" : "1762352483606",
     "passengerName" : "Willie Hernandez",
     "bagInfo" : [
       {
         "tagNum" : "17657806243915",
         "routing" : "MIA/LAX/MEL",
         "lastActionCode" : "OFFLOAD",
         "lastSeenStation" : "MEL",
         "lastSeenTimeGmt" : "2019.03.13 at 22:10:00 EDT",
         "flightLegs" : [
           {
             "flightNo" : "BM604",
             "flightDate" : "2019.03.12 at 20:00:00 EDT",
             "fltRouteSrc" : "MIA",
             "fltRouteDest" : "LAX",
             "estimatedArrival" : "2019.03.12 at 23:00:00 PDT",
             "actions" : [
               { "actionCode" : "ONLOAD to LAX",
                 "actionTime" : "2019.03.12 at 21:14:00 EDT" },
               { "actionCode" : "BagTag Scan at MIA",
                 "actionTime" : "2019.03.12 at 20:48:00 EDT"
               },
               { "actionCode" : "Checkin at MIA",
   ```

```
              "actionTime" : "2019.03.12 at 19:50:00 EDT"
            }
          ]
        },
        {
          "flightNo" : "BM667",
          "flightDate" : "2019.03.12 at 23:14:00 PDT",
          "fltRouteSrc" : "LAX",
          "fltRouteDest" : "MEL",
          "estimatedArrival" : "2019.03.13 at 22:16:00 EDT",
          "actions" : [
            { "actionCode" : "Offload to Carousel at MEL",
              "actionTime" : "2019.03.13 at 22:14:00 EDT"
            },
            { "actionCode" : "ONLOAD to MEL",
              "actionTime" : "2019.03.13 at 03:35:00 PDT"
            },
            { "actionCode" : "OFFLOAD from LAX",
              "actionTime" : "2019.03.13 at 03:19:00 PDT"
            }
          ]
        }
      ]
    }
  ]
}
```

The following index can be created on the baggage table.

```
create index idx on baggage (
  bagInfo[].lastSeenStation as string,
  bagInfo.flightLegs[].fltRouteSrc as string,
  bagInfo.flightLegs[].flightDate as string)
```

The index indexes 3 fields of the baggage table: one is the lastSeenStation field of the outer bagInfo array and the other two are the fltRouteSrc and flightDate fields of the nested flightLegs array. The contents of this index for the sample row in baggage are:

```
  [ "MEL", "MIA", "2019.03.12 at 20:00:00 EDT", "1762352483606" ]
  [ "MEL", "LAX", "2019.03.12 at 23:14:00 PDT", "1762352483606" ]
```

The following query returns the number of bags that traveled from "MIA" to "MEL" during the whole year 2019. All of the conditions in the WHERE clause are pushed to the above index, as start/stop predicates.

```
select count(b.info.bagInfo.flightLegs[$element.fltRouteSrc ="MIA"])
    from baggage b
    where exists b.info.bagInfo[$element.lastSeenStation = "MEL"].
        flightLegs[$element.fltRouteSrc = "MIA" and
        $element.flightDate >= "2019.01.01" and
        $element.flightDate <= "2019.12.31"]
```

[KVSTORE-271]

2. Added a parameter to control the maximum number of index keys generated per row. The parameter is called rnMaxIndexKeysPerRow and its default value is 10,000. When a row is being indexed (either for the first time or after an update), if the number of index keys extracted from the row exceeds the maximum, an IllegalArgumentException will be thrown.

   [KVSTORE-978]

3. Added the following five new SQL functions to extract row properties that are not stored as table columns. Although the signature of these functions specifies AnyRecord as the type of the input parameter, the functions actually require a row as input. The only expression that returns a row is a row variable, that is, a table alias whose name starts with "$".

   integer partition(AnyRecord)
   Returns the partition the row belongs to.

   integer shard(AnyRecord)
   Returns the shard that contains the row currently.

   integer row_storage_size(AnyRecord)
   Returns the storage size (in bytes) of the most recent version of the row. The returned size includes any overhead (bookkeeping) bytes. It does not include the storage size of any older versions of the row (which are now obsolete and subject to removal by the storage engine cleaner module).

   integer index_storage_size(AnyRecord, string)
   The second argument to this function is supposed to be the name of an index on the table containing the input row. The functions returns the storage size (in bytes) of the index entry or entries that "point" back to the input row from the given index. The returned size includes any overhead (bookkeeping) bytes associated with these entries.

   Notice: for performance reasons it is recommended that queries do not contain multiple calls to the index_storage_size function, for different indexes.

   timestamp(3) modification_time(AnyRecord)
   Returns the most recent modification time of the row, as a timestamp value of precision 3 (milliseconds). If the row has never been modified since its insertion, it returns the insertion time.

   Here is an example query that returns, for each partition, the total number of bytes used to store all the rows of table "foo" contained in that partition:

   ```
   select partition($f), sum(row_storage_size($f))
   from foo $f
   group by partition($f)
   ```

   [KVSTORE-127]

4. Added SQL function parse_json(string). It converts a string argument, which is supposed to be JSON text, to an Oracle NoSQL value that represents the given JSON. Here is a usage example:

   ```
   create table foo (id integer, jcol json, primary key(id))
   ```

```
insert into foo values (10, parse_json("{ \"name\" : \"john\", \"age\" :
30 }"))
```

5. Support LEFT OUTER JOIN SQL syntax.

   The NESTED TABLES clause is equivalent to a number of left-outer-join operations "centered" around the target table, this feature supports LEFT OUTER JOIN(LOJ) syntax to be compatible with standard ANSI SQL. The functionality of LOJ is subset of that of NESTED TABLES, it supports to query linear nested tables only. Here is an example:

   ```
   create table A (ida integer, a1 string, primary key(ida));
   create table A.B (idb integer, b1 string, primary key(idb));
   create table A.B.C (idc integer, c1 integer, primary key(idc));

   select * from A a LEFT OUTER JOIN A.B b ON a.ida = b.ida
                     LEFT OUTER JOIN A.B.C c ON b.ida = c.ida and b.idb =
   c.idb
   ```

   [KVSTORE-265]

6. Support MRCounter for integer, long and number value types in multi-region tables. It's a data structure that can be replicated across regions, where replicas are updated independently and can always converge to a correct common state.

   The syntax to create a MRCounter column is:

   ```
   AS MR_COUNTER
   ```

   Such columns can only be updated using + or - operators in UPDATE DML.

   [KVSTORE-803]

Bug and Performance Fixes

1. Changed the Streams API such that by default it would reconnect infinitely when the connection to any shard in source store is dropped, till the connection is back or the user shuts down the stream. Today the Streams API would reconnect up to a limit before shutting down the stream. The previous behavior favors consistency while the new behavior favors availability over consistency. The Streams API user can override the default behavior by the public API NoSQLSubscriptionConfig.Builder.setMaxReconnect().

   [KVSTORE-726]

2. Changed the Streams API that only durable write operations will be streamed. By durable write it means only the writes satisfying the durability setting. For example, if a write operation comes with durability setting COMMIT_SYNC, it will be stream only after it has been acknowledged by majority of replicas. Previously, such write may be streamed before it is acknowledged by majority of replicas, and when master migrates, the streamed operation may be missing from the new master.

   [KVSTORE-686]

3. Fixed a bug that when the user drops a region with pending DDLs to create, alter or drop a multi-region tables from that remote region, the region agent serving that remote region may not shut down correctly. Consequently if the user creates multi-region tables again immediately after the region is dropped, the XRegion service may not be able to work properly to serve the newly created multi-region tables.

   [KVSTORE-901]

4. Fixed a bug that when all subscribed tables are unsubscribed from a stream to create an empty stream, and later the user subscribes tables to the empty stream, the Stream client may throw NullPointerException.

   [KVSTORE-904]

5. Improved javadoc for asynchronous execution methods. We added the "Thread Model for Asynchronous Execution" section in the KVStore interface to describe our thread model as well as the requirement for user-supplied actions triggered after the asynchronous execution.

   [KVSTORE-938]

6. GC tuning for more predictable latencies. The following changes are made to the RN's GC parameters:

   - -XX:G1MixedGCCountTarget to 12 from 32.
   - -XX:ConcGCThreads is set to the same values as -XX:ParallelGCThreads.

   These changes should make full GC pauses less frequent. There may be a throughput degradation of around 5%, depending on the workload.

   [KVSTORE-888]

7. Fixed a bug where an httpproxy instance running on a 19.3-based client library communicating with a 20.2-based server might fail to execute queries.

   [KVSTORE-833]

8. Query processor will now always raise an error if the search-geometry argument to the geo search functions is not a valid geometry. Until now, an error would be raised only if it could be detected at compile time that the argument was invalid. If not, then during runtime the geo search functions would return false in this case.

   [KVSTORE-805]

9. Fixed a bug that would cause an IllegalArgumentException to be raised due to imprecise computation of the return type of conditional array constructor expression. For example, IAE would be raised for this query: select arr[] from Foo where "arr" is a column of table Foo with type ARRAY(ARRAY(integer)). This is because the arr[] expression in the SELECT is wrapped into a conditional array constructor whose type would be ARRAY(JSON) and it is not allowed to insert a value of type ARRAY(integer) into an array of type ARRAY(JSON). Now, the type of the conditional array constructor is set to ARRAY(ANY).

   [KVSTORE-968]

10. Because support for the oracle.kv.Consistency.NONE_REQUIRED_NO_MASTER consistency policy has been deprecated, the Oracle NoSQL Hadoop/Hive/BigDataSQL integration no longer supports that policy. Applications that employ those integrations should use the oracle.kv.Consistency.NONE_REQUIRED consistency policy instead.

    [KVSTORE-837]

11. The socket read timeout, which can be changed by calling KVStoreConfig.setSocketReadTimeout, no longer needs to be greater than the request timeout when using the async network protocol, which is enabled by default.

    [KVSTORE-776]

12. Fixed a bug in GROUP BY query based on full table scan that returns unexpected results with specified size limit. The bug exists only on cloud environment.

[NOSQL-338]

13. Fixed a bug that would cause an exception if a bind variable were used in an ADD clause of an UPDATE statement. This is when the bind variable is the new element to add to the target array and there is no position expression. For example:

```
declare $userid integer;
update teams t
add t.info.teams[1].userids $userid
where id = 1
returning *
```

[KVSTORE-963]

14. Increased the sizes of the default values of several debug and GC log parameters so that more information is retained, to help when debugging problems. Also, to save space, disabled JE debug logging by default, since the entries in those files are present in the associated service debug log files.

    Note that default parameter values are only used for new stores and new services. If you are upgrading an existing store, we recommend changing these parameters for existing services.

    Parameters that were changed:

    rnGCLogFileSize
    Changed the default file size limit for GC log files on RNs from 1048576 bytes (1 MB) to 5242880 bytes (5 MB)

    adminLogFileLimit
    Changed the default file size limit for debug log files on admins from 4000000 bytes to 5242880 bytes (5 MB)

    GCLogFileSize
    Changed the default file size limit for GC log files on admins from 1048576 bytes (1 MB) to 5242880 bytes (5 MB)

    serviceLogFileLimit
    Changed the default file size limit for other debug log files, including the store wide log, stat file, perf file, and RN debug log files, from 2000000 bytes to 5242880 bytes (5 MB)

    loggingConfigProps
    Set the default value of the logging configuration for RNs and admins to com.sleepycat.je.util.FileHandler.level=OFF, which disables logging to JE debug log files

    [KVSTORE-878]

15. Fixed a bug in master rebalancing to avoid futile retries when the capacity of the SN is not a multiple of the RF.

    [KVSTORE-869]

16. Fixed a bug that the final "Z" was missing from the timestamp string in ISO8601 format that indicates UTC.

    [KVSTORE-801]

17. If a prepared proxy-based query was executed after its table was drooped, an NPE would be raised. Now, a QueryException is raised.

[KVSTORE-920]

18. Fixed a bug that would cause an exception if a query compared a timestamp field with a bind variable and the timestamp field is indexed.

    Also allow a timestamp index to be used by a query that compares a timestamp field with a timestamp value when the precisions of the timestamp field and the value are different.

    [KVSTORE-950]

19. Fixed a rare problem where write requests could be sent to a RepNode that was in the UNKNOWN state and was not the master. In this case, the request would fail, but would have succeeded if it had been sent to the actual master. The client now gets the correct node status information and sends the request to the master.

    [KVSTORE-681]

20. Modified the rnRHAsyncMaxConcurrentRequests RepNode parameter to require that RNs be restarted when these parameters are changed. The facilities needed to support changing this value without requiring a restart had a negative impact on performance.

    [KVSTORE-941]

21. Both the JSON and text output of the ping command contain a new boolean attribute isMasterBalanced. If the value is false, that means there is an excess of RepNodes in the Master state and one or more of the nodes in the Master state will transition to a Replica at a suitable time in the future.

    [KVSTORE-799]

22. The ping command now exits with exit code 103 when the store is operational but some RepNodes are in the UNKNOWN state.

    [KVSTORE-756]

23. The output of the ping command now includes information about the storage type for RepNodes and Admins. The possible storage types reported are:

    HD
    Hard Disk

    SSD
    Solid-State Drive, flash-based

    NVME
    NVM Express, non-volatile memory

    HD (default for UNKNOWN)
    the storage type is assumed to be Hard Disk because it was not possible to determine the actual storage type.

    [KVSTORE-802]

24. Fixed a bug where, in rare cases, operations may return with the following error when a client using 20.3.17 release is connecting to a store with 20.3.17 release or after:

    ```
    java.lang.IllegalStateException: Expected state BEFORE_READ, was
    DONE
        at
    ```

```
oracle.kv.impl.async.AbstractResultHandler.checkCallOnResult(AbstractResul
tHandler.java:71)
    at
oracle.kv.impl.async.AbstractResultHandler.checkCallOnResult(AbstractResul
tHandler.java:49)
    at
oracle.kv.impl.async.AbstractDelegatingResultHandler.onResult(AbstractDele
gatingResultHandler.java:49)
    at
oracle.kv.impl.async.AsyncVersionedRemoteDialogInitiator.onCanRead(AsyncVe
rsionedRemoteDialogInitiator.java:157)
    at
oracle.kv.impl.async.dialog.DialogContextImpl$OnCanReadTask.doWork(DialogC
ontextImpl.java:1028)
    at
oracle.kv.impl.async.dialog.DialogContextImpl$OnCanReadTask.run(DialogCont
extImpl.java:1017)
    at java.base/
java.util.concurrent.Executors$RunnableAdapter.call(Executors.java:515)
    at java.base/java.util.concurrent.FutureTask.run(FutureTask.java:264)
    at
oracle.kv.impl.async.dialog.nio.NioChannelExecutor.runTasks(NioChannelExec
utor.java:1493
```

[KVSTORE-880]

25. Fixed a bug where operations may return with the following error when there is a network-related problem:

```
java.lang.IllegalStateException: Unexpected null read
    at
oracle.kv.impl.async.AsyncVersionedRemoteDialogInitiator.onCanRead(AsyncVe
rsionedRemoteDialogInitiator.java:131)
    at
oracle.kv.impl.async.dialog.DialogContextImpl$OnCanReadTask.doWork(DialogC
ontextImpl.java:1078)
    at
oracle.kv.impl.async.dialog.DialogContextImpl$OnCanReadTask.run(DialogCont
extImpl.java:1049)
    at
java.util.concurrent.Executors$RunnableAdapter.call(Executors.java:511)
    at java.util.concurrent.FutureTask.run(FutureTask.java:266)
    at
oracle.kv.impl.async.dialog.nio.NioChannelExecutor.runTasks(NioChannelExec
utor.java:1655)
```

[KVSTORE-899]

26. Fixed a scalability problem for the storage nodes that when the storage nodes have a limited CPU resource and a large number of client connections, OutOfMemoryError may occur.

    [KVSTORE-390]

27. In previous versions, the syntax of the CREATE INDEX and CREATE FULLTEXT INDEX statements allowed square brackets ([<cond>?]) when indexing the values of map fields.

With this release, square brackets ([<cond>?]) are no longer allowed, and only .values(<cond>?) can be used. This new requirement exposed a bug in the integration with Elasticsearch fulltext indexing. This bug occurs when .values(<cond>?) rather than brackets is used in the CREATE FULLTEXT INDEX statement to specify that the values of a given map should be indexed. Without the fix delivered in this release, the mechanism that generates the document to index in Elasticsearch will employ the wrong mapping; which causes fulltext search queries on the values of the map to fail. This bug is now fixed; thus, when .values(<cond>?) is used in the CREATE FULLTEXT INDEX statement, such queries will now succeed.

[KVSTORE-959]

Changes in 20.3.26

The following changes were made in Oracle NoSQL Database Release 20.3.26 Enterprise Edition.

Topics

Bug and Performance Fixes

1. Fixed a deadlock issue related to SSLDataChannel. The issue will produce stack traces similar to the following in a thread dump:

```
"ReplicaOutputThread" #829 daemon prio=5 os_prio=0 cpu=8695.34ms
elapsed=66577.11s tid=0x000055f030c1f800 nid=0x49f2 waiting for monitor
entry  [0x00007fc720171000]
   java.lang.Thread.State: BLOCKED (on object monitor)
    at
com.sleepycat.je.rep.utilint.net.SSLDataChannel$ChannelWriteTask.run(SSLDa
taChannel.java:1425)
    - waiting to lock <0x00000010046df7e0> (a java.nio.HeapByteBuffer)
    at
com.sleepycat.je.rep.utilint.net.SSLDataChannel$ChannelWriteTask.access$11
00(SSLDataChannel.java:1413)
    at
com.sleepycat.je.rep.utilint.net.SSLDataChannel.flush(SSLDataChannel.java:
590)
    ......
"REPLICA rg15-rn2(2)" #68 daemon prio=5 os_prio=0 cpu=9106.65ms
elapsed=68622.76s tid=0x000055f030c55800 nid=0xb24 in Object.wait()
[0x00007fc722b93000]
   java.lang.Thread.State: WAITING (on object monitor)
    at java.lang.Object.wait(java.base@11.0.8/Native Method)
    - waiting on
    at java.lang.Object.wait(java.base@11.0.8/Object.java:328)
    at
com.sleepycat.je.rep.utilint.net.SSLDataChannel$ChannelTask.transitToInPro
gress(SSLDataChannel.java:1335)
    - waiting to re-lock in wait() <0x00000010046e01b8> (a
java.lang.Object)
    at
com.sleepycat.je.rep.utilint.net.SSLDataChannel$ChannelWriteTask.run(SSLDa
taChannel.java:1423)
    at
com.sleepycat.je.rep.utilint.net.SSLDataChannel$ChannelWriteTask.access$11
00(SSLDataChannel.java:1413)
    at
com.sleepycat.je.rep.utilint.net.SSLDataChannel.doWrap(SSLDataChannel.java
:910)
    - locked <0x00000010046df7e0> (a java.nio.HeapByteBuffer)
    at
```

```
com.sleepycat.je.rep.utilint.net.SSLDataChannel.handshake(SSLDataCha
nnel.java:847)
    at
com.sleepycat.je.rep.utilint.net.SSLDataChannel.unwrap(SSLDataChanne
l.java:771)
    at
com.sleepycat.je.rep.utilint.net.SSLDataChannel.read(SSLDataChannel.
java:403)
    ......
```

[KVSTORE-1021]

2. Fixed a livelock issue related to SSLDataChannel. The issue is observed as an inactive channel with either SSLDataChannel doWrap or doUnwrap. Stack traces similar to the following will be seen in a thread dump:

```
   java.lang.Thread.State: RUNNABLE
        at java.lang.Object.hashCode(java.base@11.0.6/Native Method)
        at
java.util.concurrent.ConcurrentHashMap.replaceNode(java.base@11.0.6/
ConcurrentHashMap.java:1111)
        at
java.util.concurrent.ConcurrentHashMap.remove(java.base@11.0.6/
ConcurrentHashMap.java:1102)
        at
com.sleepycat.je.utilint.LoggerUtils.logMsg(LoggerUtils.java:538)
        at
com.sleepycat.je.utilint.LoggerUtils.logMsg(LoggerUtils.java:445)
        at
com.sleepycat.je.rep.utilint.net.DataChannelFactoryBuilder$ChannelIn
stanceLogger.log(DataChannelFactoryBuilder.java:417)
        at
com.sleepycat.je.rep.utilint.net.SSLDataChannel$ChannelWriteTask.ru
n(SSLDataChannel.java:1636)
        - locked <0x00000000f037cc80> (a java.lang.Object)
        at
com.sleepycat.je.rep.utilint.net.SSLDataChannel$ChannelWriteTask.acc
ess$1400(SSLDataChannel.java:1613)
        at
com.sleepycat.je.rep.utilint.net.SSLDataChannel.doWrap(SSLDataChanne
l.java:990)
    ......
   java.lang.Thread.State: RUNNABLE
        at java.lang.Object.notifyAll(java.base@11.0.6/Native
Method)
        at
com.sleepycat.je.rep.utilint.net.SSLDataChannel$ChannelTask.transitT
oIdle(SSLDataChannel.java:1513)
        - locked <0x00000000f039e788> (a java.lang.Object)
        at
com.sleepycat.je.rep.utilint.net.SSLDataChannel$ChannelReadTask.run(
SSLDataChannel.java:1808)
        at
com.sleepycat.je.rep.utilint.net.SSLDataChannel$ChannelReadTask.acce
ss$1800(SSLDataChannel.java:1751)
```

```
        at
com.sleepycat.je.rep.utilint.net.SSLDataChannel.readChannelForBufUnderflo
w(SSLDataChannel.java:1168)
        at
com.sleepycat.je.rep.utilint.net.SSLDataChannel.doUnwrap(SSLDataChannel.ja
va:1079)
        - locked <0x00000000f0386f20> (a java.lang.Object)
        - locked <0x00000000f0386ea8> (a java.lang.Object)
        at
com.sleepycat.je.rep.utilint.net.SSLDataChannel.handshake(SSLDataChannel.j
ava:930)
        at
com.sleepycat.je.rep.utilint.net.SSLDataChannel.unwrap(SSLDataChannel.java
:828)
        at
com.sleepycat.je.rep.utilint.net.SSLDataChannel.read(SSLDataChannel.java:4
41)
    ......
```

[KVSTORE-1022]

3. Fixed a bug where, in rare cases, operations may return with the following error:

```
java.lang.IllegalStateException: Expected state BEFORE_READ, was DONE
    at
oracle.kv.impl.async.AbstractResultHandler.checkCallOnResult(AbstractResul
tHandler.java:71)
    at
oracle.kv.impl.async.AbstractResultHandler.checkCallOnResult(AbstractResul
tHandler.java:49)
    at
oracle.kv.impl.async.AbstractDelegatingResultHandler.onResult(AbstractDele
gatingResultHandler.java:49)
    at
oracle.kv.impl.async.AsyncVersionedRemoteDialogInitiator.onCanRead(AsyncVe
rsionedRemoteDialogInitiator.java:157)
    at
oracle.kv.impl.async.dialog.DialogContextImpl$OnCanReadTask.doWork(DialogC
ontextImpl.java:1028)
    at
oracle.kv.impl.async.dialog.DialogContextImpl$OnCanReadTask.run(DialogCont
extImpl.java:1017)
    at java.base/
java.util.concurrent.Executors$RunnableAdapter.call(Executors.java:515)
    at java.base/java.util.concurrent.FutureTask.run(FutureTask.java:264)
    at
oracle.kv.impl.async.dialog.nio.NioChannelExecutor.runTasks(NioChannelExec
utor.java:1493
```

[KVSTORE-880]

4. Fixed a scalability problem for the storage nodes that when the storage nodes have a limited CPU resource and a large number of client connections, OutOfMemoryError may occur.
[KVSTORE-390]

5. Fixed a bug in GROUP BY query based on full table scan that returns unexpected results with specified size limit. The bug exists only on cloud environment.
[NOSQL-338]

6. Fixed a bug that would cause an exception if a query compared a timestamp field with a bind variable and the timestamp field is indexed. Also allow a timestamp index to be used by a query that compares a timestamp field with a timestamp value when the precisions of the timestamp field and the value are different.
[KVSTORE-950]

7. Fixed a bug that would cause an exception if a bind variable were used in an ADD clause of an UPDATE statement. This is when the bind variable is the new element to add to the target array and there is no position expression. For example:

```
declare $userid integer;
update teams t
add t.info.teams[1].userids $userid
where id = 1
returning *
```

[KVSTORE-963]

8. If a prepared proxy-based query was executed after its table was drooped, an NPE would be raised. Now, a QueryException is raised.
[KVSTORE-920]

9. Fixed a problem where RepNode or MasterRebalanceThread in StorageNode may exit unexpectely due to uncaught InternalFaultException(s) when transient network issues occur. In this case, the logging output shows both of them exit because of SessionAccessException, which is incorrect and has also been fixed.

```
2021-02-12 10:16:26.575 UTC SEVERE [sn6] MasterRebalanceThread
thread exiting due to exception.
oracle.kv.impl.security.SessionAccessException:
java.rmi.ConnectIOException: Exception creating connection to:
...
2021-02-12 10:16:26.575 UTC SEVERE [sn6] MasterRebalanceThread
thread exiting due to exception.
oracle.kv.impl.security.SessionAccessException:
java.rmi.ConnectIOException: Exception creating connection to:
...
```

[KVSTORE-942]

10. Fixed a bug that using UPDATE statement with REMOVE clause to remove an element from ARRAY(RECORD) or MAP(RECORD) fails with java.lang.IllegalArgumentException: Error: at (...) Cannot remove fields from records.
[KVSTORE-1200]

11. Fixed a problem where a row with an homogenous JSON array of type STRING written with release 20.2 or earlier releases would fail to deserialize in release 20.3 or later releases . The issue shows up as a row that is not found in the result of a get() or in results of queries that need to deserialize the row value. The data remains intact but is not accessible.
[KVSTORE-1347]

12. Fixed a problem where a query run on a table with data that was created prior to release 19.5 and never modified might fail with an error that indicates "modification time not available." The same error might occur if the Row.getLastModificationTime() call is made on such a row.
[KVSTORE-1234]
13. Fixed a bug where an RN could crash after an upgrade due to use of the wrong version of serialization code during the upgrade.
[KVSTORE-1361]
14. Fixed a bug preventing alterations to a table with a UUID column: rows inserted before the schema evolution would not deserialize after the schema evolution.
[KVSTORE-1357]
15. Fixed a problem where a put operation on a multi-region table that uses the proxy (httpproxy) would fail with the error "The region id cannot be 0 for multi-region table" .
[KVSTORE-1331]

API Changes

1. Row.getExpirationTime() no longer throws an exception if the underlying Row metadata is not available. Instead it will return 0 indicating no expiration. Expiration time is 0 for rows that either do not have a TTL or were written prior to the introduction of the TTL feature.
[KVSTORE-1234]

Changes in 20.3.17

The following changes were made in Oracle NoSQL Database Release 20.3.17 Enterprise Edition.

Topics

New Features

1. Support Universally Unique Identifier.

   Added new SQL function random_uuid() which returns a randomly generated UUID, as a string of 36 characters.

   Added new option "AS UUID" for the STRING data type. The syntax is "STRING AS UUID". The input and output format of "STRING AS UUID" column must be UUID canonical format: 32 hexadecimal (base 16) digits, displayed in five groups separated by hyphens, in the form 8-4-4-4-12 for a total of 36 characters (32 hexadecimal characters and 4 hyphens).

   Added new option "GENERATED BY DEFAULT" for the "STRING AS UUID" data type. The syntax is STRING AS UUID GENERATED BY DEFAULT. The system automatically generates a value for the UUID column if the user does not supply a value for it.

   [KVSTORE-215]

2. New client side cache of table metadata.

   Added a new client side cache of table metadata. This affects the various TableAPI.getTable() methods. The cache will improve performance of these methods by not requiring a remote call to the server for Table instances found in the cache.

   [KVSTORE-608]

Bug and Performance Fixes

1. Fixed a problem that the Streams API may incorrectly pull a table with earliest version when it subscribes a new table. Instead, it should pull the table with latest or required version.

   [KVSTORE-492]

2. Made the region name defined in the JSON configuration file required by the XRegion agent case-insensitive.

   [KVSTORE-335]

3. Fixed a problem that the XRegion agent may fail to initialize the table when reading/writing internal info from/to the local store.

   [KVSTORE-638][KVSTORE-651]

4. Modified the default timeout of a subscription created by Streams API from 10 minutes to 30 seconds.

   [KVSTORE-660]

5. Fixed a performance bug where, during the multi-region table transfer from a remote region, the table might be copied more than once.

   [KVSTORE-483]

6. Fixed a problem where setting a mutable JE parameter for an Admin or an Arbiter Node did not have an effect unless the node was restarted explicitly.

   [KVSTORE-621]

7. Modified the implementation of multi-region tables to create checkpoints when initializing tables. If XRegion Agent fails while it is copying the contents of a multi-region table from another region, the copy can resume from a checkpoint when the agent restarts, reducing the time needed to complete initialization.

   [KVSTORE-40]

8. Plan locking has been changed to reduce failures due to concurrent plan execution. This change generally impacts system plans run by the Admin.

   [KVSTORE-432]

9. Less strict indexing in case of typed JSON indexes.

   Currently, if a JSON field is indexed as double/float and a row to be inserted contains an integer/long on that field, that insertion raises an error. This is too strict. To fix, the integer/long value is cast to double/float and if the cast is lossless, the insertion succeeds.

   [KVSTORE-571]

10. Made modifications to avoid a problem where Java RMI temporarily creates additional RenewClean threads. The additional threads were being created in both Replication Nodes and clients.

    [KVSTORE-493]

11. Fixed problems with the KVStore.executeAsync() API. Previously, this API worked for very simple queries only. For other queries, it could cause crashes or infinite loops. The API now works for all queries.

    [KVSTORE-597]

12. Some behavior of the SET LOCAL REGION DDL command has been changed. With the new behavior, the local region name cannot be changed if a multi-region table has been created. The one exception is if the store is upgraded from a version that did not support the local region name. In that case the local region name can be set once if multi-region tables exist. Lastly, though the local region name is case insensitive, case will be preserved.

    [KVSTORE-572]

13. Fixed a problem where, if a sorting/grouping query used hints to force an index and the index was not a sorting one, generic order-by/group-by was not added to the query.

    [KVSTORE-624]

14. Fixed a bug with queries that use multi-key index and aggregate function but no grouping expressions. A multi-key index can never be a sorting index. This rule was not taken into account when a query contained aggregate functions without any group-by clause. As a result, generic group by was not added as required.

[KVSTORE-626]

15. In previous releases, the integration of Oracle NoSQL Database with Apache Hadoop MapReduce, Apache Hive, and Oracle Big Data SQL was documented via javadoc package summaries included with the separate example distribution. With this release, that information is now formally documented in the following publicly available user guides:

    - Integration with Apache Hadoop MapReduce
    - Integration with Apache Hive
    - Integration with BDS

    [KVSTORE-213][KVSTORE-430][KVSTORE-431][#27588]

16. The integration of Oracle NoSQL Database with Elasticsearch so that full text search queries can be performed on data stored in an Oracle NoSQL Database table is now formally documented in the following publicly available user guide:

    - Integration with Elastic Search for Full Text Search

    [#27380]

17. Fix problems that could cause calls made using the asynchronous table API (for example TableAPI.getAsync) on a secure store to fail if an authentication token could not be renewed and reauthentication was needed.

    [KVSTORE-727]

18. Fixed a problem where adding an empty row to a table with an identity column would result in an exception: "IllegalArgumentException: Primary key is empty".

    [KVSTORE-502]

19. Fixed a bug where "get kv" failed when used with multi-region tables.

    [KVSTORE-615]

Storage Engine Changes (JE 20.3)

1. If the master in a replication group is about to fail with a DiskLimitException it will first check if there are any lagging replicas that are preventing it from from deleting logs. If a lagging replica is found, the master will break the connection with the replica, forcing the replica to reconnect and perform a NetworkRestore.

    [KVSTORE-145](20.3.0)

2. Several fixes were made to Btree/data verification.

    - When an invalid LSN for an IN (as opposed to an LN) is discovered, the verifier is designed to abort verification of the current database and move to the next database. Previously, the verifier was not aborting verification properly and was reporting an internal exception (OperationVerifyException) along with potentially large numbers of NullPointerExceptions before moving on to the next database. These errors were logged and reported as VerifyError.Problem.RECORD_ACCESS_EXCEPTION. This has been fixed so verification does abort properly and these exceptions do not occur.
    - When verification is run via DbVerify, logging of individual problems is not enabled because the JE environment is read-only. To cause output to System.err, the java.util.logging.config.file JVM property may be set to the name of a file containing com.sleepycat.je.util.ConsoleHandler.level=ALL .Previously, because rate-

limited logging is used by the verifier, when explicitly enabling logging as described above, only a small subset of the individual problems were logged. Now, rate-limited logging is disabled when DbVerify is used. This ensures that all problems are logged for debugging purposes, but may produce a large amount of output.

- The javadoc for DbVerify, VerifyConfig and RateLimitingLogger has been improved.

[KVSTORE-616](20.3.4)

3. Fixed a bug that could cause DbVerify to report false VerifyError.Problem.DATABASE_ACCESS_EXCEPTION errors, and other utilities and programs that open the environment as read-only to report false EnvironmentFailureException.UNEXPECTED_STATE errors.

   [KVSTORE-706](20.3.9)

4. CkptStart and CkptEnd log records will now be printed with a timestamp using UTC instead of the local time. This is the format used for all other log records.

   [KVSTORE-627](20.3.13)

Changes in 20.2.17

The following changes were made in Oracle NoSQL Database Release 20.2.17 Enterprise Edition.

Topics

- Bug and Performance Fixes

Bug and Performance Fixes

1. Updated 3rd party libraries to current versions.
2. In extreme situations, when an RN is unable to find a suitable feeder for a Network Restore because all other RNs are down or unreachable, the RN gives up, but does not exit the process. This can result in the environment never being restored and the RN process is effectively hung, unable to process requests and needs to be explicitly killed and restarted. The behavior has been changed so that the RN exits and the SNA restarts it with a clean process state with which to retry the Network Restore.

   [KVSTORE-610]
3. A SessionAccessException in the SNA resulted in the SNA halting the Master Balancing functionality at the SN, as indicated by the following representative entries in the SNA's logs:

   ```
   ...
   2020-08-10 00:27:33.875 UTC SEVERE [sn5] MasterRebalanceThread
   thread exiting due to exception.
   ...
   2020-08-10 00:27:33.878 UTC INFO [sn5] Master balance manager
   shutdown
   2020-08-10 00:27:33.878 UTC INFO [sn5] MasterRebalanceThread thread
   exited.
   ...
   ```

   The SNA now handles the SessionAccessException and retries the operation.

   [KVSTORE-595]
4. Fixed a problem where setting a mutable JE parameter for a Replication Node did not have an effect unless the node was restarted explicitly. This problem was introduced in release 20.2.16.

   [KVSTORE-574]
5. Fixed a problem where an unresolved network address for the host of a storage node could cause requests to fail with RequestTimeoutException. Once the problem occurred, it was persistent, and all future client calls would timeout until the client was restarted. If client logging was enabled, the logging output might include an exception like the following:

   ```
   2020-08-12 13:09:40.047 UTC SEVERE - Uncaught exception in
   thread:KV c-5287366135758798770 RepNodeStateUpdateThread_Updater_1
   java.nio.channels.UnresolvedAddressException
   ```

```
    at
oracle.kv.impl.async.AbstractCreatorEndpoint.startDialog(AbstractCreatorEn
dpoint.java:87)
    at
oracle.kv.impl.async.AsyncVersionedRemoteInitiator.startDialog(AsyncVersio
nedRemoteInitiator.java:131)
    at
oracle.kv.impl.async.AsyncVersionedRemoteInitiator.getSerialVersion(AsyncV
ersionedRemoteInitiator.java:102)
    at
oracle.kv.impl.async.registry.ServiceRegistryInitiator.getSerialVersion(Se
rviceRegistryInitiator.java:53)
    at
oracle.kv.impl.async.AsyncVersionedRemoteAPI.computeSerialVersion(AsyncVer
sionedRemoteAPI.java:111)
    at
oracle.kv.impl.async.registry.ServiceRegistryAPI.access$000(ServiceRegistr
yAPI.java:47)
    at
oracle.kv.impl.async.registry.ServiceRegistryAPI$Factory.wrap(ServiceRegis
tryAPI.java:152)
    at
oracle.kv.impl.util.registry.AsyncRegistryUtils.getRegistry(AsyncRegistryU
tils.java:354)
    at
oracle.kv.impl.util.registry.AsyncRegistryUtils.getRequestHandler(AsyncReg
istryUtils.java:314)
    at
oracle.kv.impl.api.rgstate.RepNodeState$AsyncReqHandlerRef.getRequestHandl
er(RepNodeState.java:1179)
    at
oracle.kv.impl.api.rgstate.RepNodeState$AsyncReqHandlerRef.resolveInterna
l(RepNodeState.java:1138)
    at
oracle.kv.impl.api.rgstate.RepNodeState$AsyncReqHandlerRef.resolve(RepNode
State.java:1083)
    at
oracle.kv.impl.api.rgstate.RepNodeState.resolveReqHandlerRef(RepNodeState.
java:251)
    at
oracle.kv.impl.api.rgstate.UpdateThread$ResolveHandler.run(UpdateThread.ja
va:322)
    at java.base/
java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:
1128)
    at java.base/
java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java
:628)
    at java.base/java.lang.Thread.run(Thread.java:834)
```

[KVSTORE-523]

Changes in 20.2.16

The following changes were made in Oracle NoSQL Database Release 20.2.16 Enterprise Edition.

Topics

New Features

1. Modified the Hive/Big Data SQL integration mechanism to support querying table fields of the data types FieldDef.Type.JSON, FieldDef.Type.NUMBER, and FieldDef.Type.TIMESTAMP. Along with this feature, new example code is also provided to help create tables with fields of these new types.

   [#25802]

2. Support Time to Live (TTL) in the Streams API and in Multi-Region Tables. The Rows supplied by the Streams API to subscribers will include the TTL expiration time for the operation from source store. Note that the expiration time is computed from the original TTL at source store where and when the operation is made, instead of the time when Streams API receives the operation. Therefore, the operation that Streams API returns may have already expired if its expiration time has passed. For Multi-Region tables, the TTL expiration time of each row is replicated to other regions, and thus a row in Multi-Region table in any region will expire at the expiration time replicated from the region where and when operation is made.

   We strongly recommended upgrading Multi-Region agents and stores in all regions to this release before specifying any rows with non-zero TTL values for Multi-Region tables. Expiration times will be lost when rows are replicated to regions running older software versions, meaning that the copied rows will not expire.

   [#28165]

3. Make XRegion Service agent parameter configurable in JSON config file. This release adds following configurable parameters with their default values in the JSON config file:

   ```
   {
     "checkpointIntvSecs" : 300,
     "checkpointIntvOps" : 1048576
   }
   ```

   [#28152]

4. Implemented generic ORDER BY and GROUP BY.

   In previous releases, ORDER BY and GROUP BY were possible only if there was an index that sorted the rows by the ORDER BY/GROUP BY expressions. Furthermore, it was not possible for queries to include both ORDER BY and

GROUP BY. These restrictions are lifted in this release. For example, the following query returns the monthly sales for year 2020, ordered by the sales amount.

```
select f.xact.month, sum(f.xact.amount) as sales
    from Foo f
    where f.xact.year = 2020
    group by f.xact.month
    order by sum(f.xact.amount);
```

Notice that generic ORDER BY and GROUP BY may consume a lot of driver memory, because of the need to materialize the full query result in driver memory. In contrast, index-based ORDER BY/GROUP BY exploit the row sorting provided by the index to avoid the materialization and caching of any intermediate results. As a result, it is recommended that users create appropriate indexes for use in ORDER BY/GROUP BY queries. For example, if there is an index on both xact.year and xact.month, the above query will use that index and the GROUP BY will be an index-based one. If no such composite index exists, but instead there are two separate indexes on xact.year and xact.month respectively, the above query will use the index on xact.year and the GROUP BY will be a generic one. This is because, in general, indexes that "cover" query predicates are preferred over indexes that "cover" ORDER BY/GROUP BY expressions.

For generic ORDER BY/GROUP BY, applications can specify (via a query-level execution option) how much memory such operations are allowed to consume at the driver, and a query will raise an exception if the maximum allowed threshold is exceeded.

[#28202]

5. Implemented SELECT DISTINCT.

   The DISTINCT keyword is now supported in the SELECT clause. If present, duplicate results will be removed from query result set. As with generic ORDER BY and GROUP BY, SELECT DISTINCT needs to cache the full result set in driver memory. The same execution option controls how much memory such operations are allowed to consume at the driver.

   [#28202]

6. Introduced a new Admin CLI command `show mrtable-agent-statistics` to show the latest statistics for Multi-Region table agents.

   ```
   show mrtable-agent-statistics [-agent <agentID>]
   [-table <tableName>] [-merge-agents]
   ```

   With no argument, the command shows combined statistics over all tables for each agent. The `-agent` flag limits the statistics shown to the agent with the specified agent ID. The `-table` flag limits the statistics shown to the Multi-Region table with the specified name. The `-merge-agents` flag combines statistics over all agents.

   [#28155]

7. Key distribution statistics enabled by default.

   The collection of key distribution statistics is now enabled by default. You can control whether statistics are collected by setting the `rnStatisticsEnabled` parameter.

   [KVSTORE-155]

8. Key distribution statistics include storage size information by default.

The collection of key distribution statistics now, by default, collects information about the storage size of the associated table entries. This information can be used to find the storage size for a table in specific partitions and across the store as a whole.

The new `statsIncludeStorageSize` parameter controls whether the collection of key distribution statistics includes information about entry storage sizes.

[KVSTORE-375]

9. Updated the `topology change-repfactor` Admin CLI command to support reducing the replication factor for online secondary zones. You can now use this command to reduce the replication factor for a secondary zone that you want to keep, or in preparation for removing the zone. To remove a secondary zone, use `topology change-repfactor` to change the replication factor of the zone to zero, and then use the `plan remove-sn` and `plan remove-zone` commands to remove the storage nodes and complete the removal of the zone. Note that you can only reduce the replication factor of a zone whose storage nodes and replication nodes are currently online.

   [KVSTORE-185]

10. In the `regex_like` function, the source param is now always implicitly cast to a string.

    [KVSTORE-342]

Bug and Performance Fixes

1. Fixed a bug which could cause OutOfMemoryErrors in service processes when there are many new client connections established during a short time. The problem is more likely for secure stores. It also appears more often on SNs, since they usually have smaller heap sizes. A heap dump from the process with such problem will show a large number of SSLDataChannel and ConnectTimeoutTask.

   [KVSTORE-288]

2. Fixed a bug that the XRegion Service agent may be blocked when a multi-region table is not found at remote regions. If any Multi-Region table is missing, the table polling thread will be created if not exist, and periodically poll the remote regions for missing tables till the table is found, and the thread will start initialization and include it in the running stream.

   [#28218] [#28194]

3. Fixed incomplete table initialization bug.

   When the Multi-Region agent restarts from a crash, there was a problem that incomplete table initialization might not resume correctly, resulting in table corruption in some cases. The fix ensures that when the Multi-Region agent restarts, the agent will reinitialize all tables whose initialization was not complete before the crash.

   [KVSTORE-321]

4. Fixed bug in min/max aggregate functions.

   In previous releases, these functions were non-deterministic if their input were values of different types. For example, if you had min(t.a.b) and t.a.b was a number in some rows and a string in others, the result would be either the min number or the min string, depending on which row was encountered first. For ORDER BY, we already defined an ordering among otherwise non-comparable

values. This ordering is now used for min/max as well, thus making their results deterministic.

[#28202]

5. Fixed a bug that can cause a query that has both index-based group-by and joins to miss one result per partition/shard.

   [KVSTORE-276]

6. When running Big Data SQL queries on tables mapped to different tables in the Oracle NoSQL store, the split information for each table is now cleared in the storage handler before executing the query. Before this fix, errors could occur in the Oracle DB when split information from a previous query on a different table is inconsistent with the split information for the current table.

   [KVSTORE-372]

7. Update security DDL commands to work with bootstrap admin without deploying any storage nodes. Before this fix, security DDL commands, such as CREATE USER, GRANT and etc, only can be executed after a storage node is deployed. You can now use security DDL command to create the initial admin user instead of deprecated `plan create-user` right after configure the store name.

   [KVSTORE-382]

8. Fixed a bug that could cause queries on Multi-Region tables to not work when the region id was greater than 119.

   [KVSTORE-418]

9. Fixed a problem which could cause a Replication Node to create a large number of threads while shutting down after encountering a fatal error. In some cases, the large number of threads created could cause other processes on the same host to fail because the maximum processes limit was reached.

   [KVSTORE-405]

10. Fixed bugs in `show tables` command on Admin CLI and sql CLI.

    If global namespace of CLI is set to `sysdefault`, `show tables` is expected to return all tables in both default and non-default namespaces. If the global namespace of CLI is set to a non-default namespace, `show tables` is expected to return those tables in the specified namespace.

    Before the fix, `show tables` on admin CLI returns no table if global namespace is `sysdefault`. On sql CLI, the global namespace was ignored, `show tables` always returns all tables.

    [#28195]

11. Fixed a bug that `show snapshots` return empty when kvstore uses customized admin storage directory.

    [KVSTORE-326]

12. Fixed a bug that `show indexes` on admin CLI does not return the index in the case where child table has index but there is no index on its parent table.

    [KVSTORE-394]

13. Fixed a bug so RNs are not restarted if a mutable JE parameter is changed.

    [KVSTORE-1]

14. Fixed a bug that the put methods in TableAPI threw a `FaultException` when the table has an identity column that ran out of values. The exception could cause unexpected retries and the request time out. Now an `IllegalArgumentException` will be thrown when an identity column reaches the limit.

    [KVSTORE-261]

15. Fixed a bug that the TableAPI could throw an `IllegalArgumentException` when the Multi-Region table service agent merges remote rows for key-only tables. The exception would eventually cause the agent terminate the stream and log the error.

    [KVSTORE-253]

16. Fixed a bug where modifying a multi-region table using an `SQL UPDATE` statement could cause a Replication Node to crash when the Multi-Region agent processed table data from a remote store. The exception that caused the crash was `IllegalArgumentException: This is not a record of multiregion tables.`

    [KVSTORE-463]

Changes in 20.1.20

The following changes were made in Oracle NoSQL Database Release 20.1.20 Enterprise Edition.

Topics

Bug and Performance Fixes

1. Fixed a bug where a LOG_FILE_NOT_FOUND exception could be generated. If TTL is used and some rows are using TTL in hours and others in days, or there is an update to the TTL with a unit change from hours to days, then it is possible to get the exception after the rows get deleted. To recover from this exception, a restore was needed.

   [KVSTORE-406]

2. Fixed a problem where setting the DNS cache TTL parameters from makebootconfig can result in a violation warning when running "verify configuration" command from the admin. The issue is that the SN correctly adopts the set TTL value but the admin is not updated during SN deployment resulting in the violation warning on the discrepancy. The warning looks like the following:

   ```
   Verify: sn1: Mismatch between metadata in admin service and sn1:
       Parameters in Admin database but not from configuration for
       service sn1: dnsCacheTTL=10
   ```

   [KVSTORE-373]

3. Fixed a compatibility problem. The problem would occur when upgrading from 19.5 release to a 20.1 release prior to this patch due to an incompatible change on the EndpointGroupStats object. When the problem occurs during the upgrade, the admin would observe a NullPointerException and a stack trace that looks like the following:

   ```
   2020-06-02 09:23:12.398 UTC SEVERE - [admin1] Collector:
       exception when polling agentId:
       rg1-rn3 Exception: java.lang.NullPointerException
   at oracle.kv.impl.measurement.EndpointGroupStats.getFormattedStats(
       EndpointGroupStats.java:95)
   at oracle.kv.impl.measurement.EndpointGroupStats.getFormattedStats(
       EndpointGroupStats.java:88)
   at oracle.kv.impl.monitor.views.PerfView.displayConciseStats(
       PerfView.java:172)
   at oracle.kv.impl.monitor.views.PerfView.applyNewInfo(PerfView.java:136)
   ```

   [KVSTORE-368]

4. Fixed a problem where delete operations could deadlock with concurrent delete requests on the same key.

   [KVSTORE-272]

5. Added the new parameters javaRnParamsOverride, javaAnParamsOverride, and javaAdminParamsOverride to allow users to alter the JVM command line parameters for

Replication Nodes, Arbiter Nodes, and Admin Nodes. Users can set these parameters for individual services, and can also make policy settings to apply to future instances of the associated service types. Going forward, users should set these new parameters rather than the javaMiscParams parameter, which should now be considered reserved for system use, although the old parameter is still supported for compatibility in old stores.

[KVSTORE-277]

6. Fixed an incompatibility with the 20.1 client when used with earlier version stores. This bug would cause some table related calls to fail with a java.lang.UnsupportedOperationException.

   [KVSTORE-416]

Changes in 20.1.16

The following changes were made in Oracle NoSQL Database Release 20.1.16 Enterprise Edition.

Topics

Bug and Performance Fixes

1. Bug fix for IN operator. When the non-standard version of the IN operator (the one that has a single expression in the right-hand-side) is used and the left-hand-side is a primary key, the compiler would worngly determine that the query is a SINGLE_PARTITION query, instead of ALL_PARTITIONS. As a result, the query will be sent to a single partition, and any results from other partitions would be lost. For example, the bug applies to the following query, if "id" is the primary key column of table Foo:

   ```
   declare $arr array(int);
   select * from Foo where id in $arr[]
   ```

   [#28145]

2. Fixed a bug in INSERT statement, when the DEFAULT keyword is used as the value of an IDENTITY column. The INSERT statement would throw an exception in this case. The bug exists only when the proxy is used.

   [NOSQL-224]

Changes in 20.1.15

The following changes were made in Oracle NoSQL Database Release 20.1.15 Enterprise Edition.

Topics

New Features

1. Added IN operator in SQL for Oracle NoSQL. The IN operator is essentially a compact alternative to a number of OR-ed equality conditions. For example, the query

```
SELECT * FROM Foo WHERE a IN (1, 5, 4)

is equivalent to

SELECT * FROM Foo WHERE a = 1 OR a = 5 OR a = 4
```

   and the query,

```
SELECT * FROM Foo
    WHERE (a, b) IN ((1, "a"), (5, "g"), (4, "t"))

is equivalent to

SELECT * FROM Foo
    WHERE (a = 1 AND b = "a") OR (a = 5 AND b = "g")
    OR (a = 4 AND b = "t")
```

   However, in addition to being more compact, queries using IN operators will be executed more efficiently if appropriate indexes exist. For example, if table Foo has an index on columns a and b, then both of the above IN queries will use that index to find the qualifying rows, whereas the equivalent OR queries will be executed via full table scans.

   The above IN queries follow the standard SQL syntax. An alternative syntax has also been implemented, that allows a relative large number of search keys to be provided via a single bind variable. For example, if the $keys variable is bound to the array [1, "a", 5, "g", 4, "t"], then the following query is equivalent to the second IN query above.

```
DECLARE $keys array(json);
    SELECT *
    FROM Foo
    WHERE (a, b) IN $keys[]
```

   [#27900]

2. Support for "untyped" json indexes.

   So far, indexing a field inside json documents required the specification of a concrete json atomic type for the field (one of integer, long, double, number, string, or boolean). If, for some document, the index field did not conform to its declared type, index creation or the insertion of that document in a table would fail. In this release, anyAtomic is added as a valid type for indexed json fields. A field indexed with the anyAtomic type can be of any valid json atomic type and the index will store all the heterogeneous values of that field.

   [#27989]

3. Query execution plans are now being displayed in json format.

   [#26016]

4. Added the new rnRHAsyncMaxConcurrentRequests parameter, which specifies the maximum number of concurrent asynchronous requests that a Rep Node can handle before it should do throttling. This parameter replaces the rnRHAsyncExecQueueSize parameter, which is now ignored, since the queue size is now computed from the difference between the rnRHAsyncMaxConcurrentRequests and rnRHAsyncExecMaxThreads parameters.

   [#27823]

5. The SQL command "ALTER TABLE" now supports adding or removing regions from existing multi-region tables.

   [#28027]

6. The new SQL command "SET LOCAL REGION" can be used to specify the name of the local region for use with multi-region tables.

   [#28013]

7. When creating a multi-region table or adding new regions to an existing multi-region table, the associated tables in remote regions no longer have to be empty.

   [#28120]

8. Multi-Region tables are now only supported in the Enterprise Edition.

   [#28144]

Bug and Performance Fixes

1. When upgrading from a version prior to 18.3, the store must be fully upgraded before an elasticity plan that includes migrating partitions can be run. If the store is not fully upgraded to 18.3 or above, a partition migration may fail, causing the elasticity plan to exit. The plan can be re-run once the store is fully upgraded. This new behavior avoids a deadlock risk between the deploy topology plan and an internal Admin plan.

   [#28036]

2. Added the snapshotName to the JSON output of the snapshot command result.

   [#27700]

3. Fixed a problem where the incorrect math context might be used in queries that use sum() and seq_sum() functions on the Number data type.

   [#28040]

4. Fixed a bug affecting on-prem, proxy-based, join queries. In this mode, some join queries would wrongly raise exception saying that a single results consumes more bytes than the max allowed.

[#28103]

5. Fixed a problem that could occur when using the HTTP proxy and drivers on queries with a LIMIT clause. It was possible that some results within the limit would be omitted, but some outside of the limit included.

   [#28034]

6. Fixed problems when the Java client is used to access multiples stores which have the same name. Previously, for secure stores, client access to the store could fail with a certificate signature validation error. In addition, socket open and read timeout values specified for a particular store were not applied correctly. Both problems are now fixed.

   [#27952]

7. TableAPI.getTable() is now more reliable when one or more nodes is unavailable.

   [#28150]

8. Made scalability improvements to reduce the overhead of using large numbers of tables and indexes.

   [#27204]

Changes in 19.5.30

The following changes were made in Oracle NoSQL Database Release 19.5.30 Enterprise Edition.

Topics

Bug and Performance Fixes

1. Fixed a bug where a LOG_FILE_NOT_FOUND exception could be generated. If TTL is used and some rows are using TTL in hours and others in days, or there is an update to the TTL with a unit change from hours to days, then it is possible to get the exception after the rows get deleted. To recover from this exception, a restore was needed. *[KVSTORE-406]*
2. Fixed a problem where setting the DNS cache TTL parameters from makebootconfig can result in a violation warning when running "verify configuration" command from the admin. The issue is that the SN correctly adopts the set TTL value but the admin is not updated during SN deployment resulting in the violation warning on the discrepancy. The warning looks like the following:

   ```
   Verify: sn1: Mismatch between metadata in admin service and sn1:
   Parameters in Admin database but not from configuration for service sn1:
   dnsCacheTTL=10
   ```

   [KVSTORE-373]
3. The DbVerifyLog utility will now verify the VLSN sequence. *[KVSTORE-313]*
4. Fixed a bug which could cause OutOfMemoryErrors in service processes when there are many new client connections established during a short time. The problem is more likely for secure stores. It also appears more often on SNs, since they usually have smaller heap sizes. A heap dump from the process with such problem will show a large number of SSLDataChannel and ConnectTimeoutTask. *[KVSTORE-288]*
5. Fixed a problem where delete operations could deadlock with concurrent delete requests on the same key. *[KVSTORE-272]*
6. During MasterTransfer transactions awaiting acknowledgements from Replicas will now quit waiting and return an InsufficientAcksException. This will allow the old Master to transition to a Replica faster. *[KVSTORE-137]*

Changes in 19.5.19

The following changes were made in Oracle NoSQL Database Release 19.5.19 Enterprise Edition.

Topics

- Bug and Performance Fixes

Bug and Performance Fixes

1. Fixed a couple of problems that prevented the system from working properly when using recent Java patch releases (Java versions 1.8.0_241, 11.0.6, and 13.0.2). In one case, starting the server would fail with a NullPointerException while handling a remote call, while in the other case a RemoteException was thrown saying that a remote method's interface did not implement Remote.

   [#28063]

2. Improved system behavior in a cluster that is read-only due to multiple node failures. Previously operations would retry until the request timeout has passed. This change detects errors that cause the operation to fail more quickly if it seems that retries will not help. An example is an error indicating that reauthentication may be required.

   [#28015]

3. Fixed a problem that could occur when using the HTTP proxy and drivers on queries with a LIMIT clause. It was possible that some results within the limit would be omitted, but some outside of the limit included.

   [#28034]

4. Fixed a problem where the incorrect math context might be used in queries that use sum() and seq_sum() functions on the Number data type.

   [#28040]

5. Fixed bug causing over-reporting of query byte consumption to the aggregation service. The bug occurs when a query executing at an RN would (a) scan an index more than once or (b) scan more than one index. An example of (a) is a geo-search query. An example of (b) is ALL PARTITION queries, where we try to access as many local partitions as possible (before reaching the 2MB limit). In such cases, consumption reported to the AS during earlier index scans would be reported again during subsequent scans.

   [#28081]

6. Updated the bundled HTTP proxy to a newer version.

Changes in 19.5.16

The following changes were made in Oracle NoSQL Database Release 19.5.16 Enterprise Edition.

Topics

New Features

1. Introduced Multi-Region Tables, a feature that lets users create "read-anywhere" and "write-anywhere" tables that live in multiple regions, where each region is a separate Oracle NoSQL Database store. **This is a preview release** and a general availability version will be available in a future release.

   [#27728]

2. Improved the way to configure the network name resolution cache for services. The address cache is used to resolve host names during name resolution. Two values are used to specify how long the results of successful or unsuccessful name lookups should be kept in the cache. See the Address Cache section of the Networking Properties page for more details.

   When creating a new SN, the values can be configured with the `-dns-cachettl` flag to the `makebootconfig` command. After deployment, the initial values can be overridden by changing the parameters `dnsCacheTTL` and `dnsCacheNegativeTTL` of the SN. Those two parameters are also policy parameters so that it can be set for future SN deployments. Each service shares the same parameter values with its monitoring SN. When changing the parameters for an SN, all service processes need to be restarted for the new values to take effect. Admin, RN, and arbiter services will be restarted automatically, but the SNs must be restarted manually.

   [#27660]

3. Added new asynchronous methods to the table API. Applications can use these methods to make calls without using a thread to wait for results, which can improve the efficiency of clients that make many concurrent calls.

   Clients now use a new network protocol that supports multiplexing multiple calls on the same socket, which is used to support asynchronous operation and to reduce the number of socket connections needed. New clients are only compatible with this version of the server, although the server continues to support old clients. When upgrading to this release, make sure to upgrade the store first before upgrading clients or HTTP proxies.

   New methods on TableAPI:

   - `getAsync`
   - `multiGetAsync`
   - `multiGetKeysAsync`
   - `tableIteratorAsync`
   - `tableKeysIteratorAsync`

- `putAsync`
- `putIfAbsentAsync`
- `putIfPresentAsync`
- `putIfVersionAsync`
- `deleteAsync`
- `deleteIfVersionAsync`
- `multiDeleteAsync`
- `executeAsync`

New methods on KVStore:

- `executeAsync`

New fields and methods on `KVStoreConfig`:

- `DEFAULT_NETWORK_ROUNDTRIP_TIMEOUT`
- `USE_ASYNC`
- `DEFAULT_USE_ASYNC`
- `getNetworkRoundtripTimeout`
- `setNetworkRoundtripTimeout`
- `getUseAsync`
- `setUseAsync`

New field in `KVStoreFactory`:

- `ENDPOINT_GROUP_NUM_THREADS_PROPERTY`

New interfaces:

- `oracle.kv.ExecutionSubscription`
- `oracle.kv.IterationSubscription`

Iteration methods (`TableAPI.tableIteratorAsync`, `TableAPI.tableKeysIteratorAsync`, and `KVStore.executeAsync`) are implemented using the Reactive Streams framework. Other asynchronous methods return `java.util.concurrent.CompletableFuture`.

Also added the new `table.AsyncExample` example.

[#24773]

4. Added the ServerResourceLimitException class. Instances of this class may be thrown if the server is unable to handle a request because of resource constraints.

   [#27301]

5. Added the new WRITE_SYSTEM_TABLE privilege and writesystable role to support modifications to system tables. The new privilege and role are intended to be used with the multi-region table agent. Normal users typically should not modify system tables.

   [#27830]

6. The Oracle NoSQL Hadoop/Hive/BigDataSQL integration code now works with Hadoop3, Hive2, and BigDataSQL 4.0. Because changes were made in Apache Hive that make Hive1 incompatible with Hive2, if you wish to run MapReduce jobs, Hive queries, and/or BigDataSQL queries in a Hadoop2, Hive1, and/or BigDataSQL 3.5.x environment, then you should use the 18.3 release of Oracle NoSQL Database; and use this release in Hadoop3/Hive2/BigDataSQL 4.0 environments.

   [#27898]

7. Source code for the NoSQL Database storage engine, also known as "Berkeley DB Java Edition", is now included in the Community Edition release package.

   [#27478]

8. Added the following string manipulation SQL functions:

   - `concatenation: arg1 || arg2 -> string , concat(arg1, arg2, ...) -> string?`
   - `substring(str, position[, for_substring_length] ) -> string`
   - `upper(str) -> string`
   - `lower(str) -> string`
   - `trim(from_str[, where[, trim_chars]]) -> string`
   - `ltrim(str) -> string`
   - `rtrim(str) -> string`
   - `length(str) -> integer`
   - `contains(str, contained_str) -> boolean`
   - `starts_with(str, start_str) -> boolean`
   - `ends_with(str, end_str) -> boolean`
   - `index_of(str, search [, pos]) -> integer`
   - `replace(str, search_str [, replacement_str] ) -> string`
   - `reverse(str) -> string`

   Also the result of casting a JSON null value to a string has been changed from RuntimeError to string "null".

   [#27771]

Bug and Performance Fixes

1. Fixes a bug where AbsoluteConsistency read requests could be directed to the master on the minority side of a network partition. AbsoluteConsistency read requests now require an authoritative master: one that is in contact with a quorum of replicas. The request will timeout if a suitable master is not available within the request timeout period.

   [#27785]

2. Enhanced the predicate pushdown mechanism in the Oracle NoSQL Hive integration code to work with changes made in Hive2 that cause some queries to produce incorrect results under Hive2.

   [#27898]

3. Fixed a bug in the capacity planning spreadsheet, where the number of machines required by a store could be overestimated in some cases.

   [#27939]

4. Fixed a bug where in some cases executing a DDL command which is a duplicate of a running DDL command, an Admin plan will be orphaned in the APPROVED state.

   [#27688]

5. Fixed a bug in sorting the special values (NULL, json null, and EMPTY) when the direction of the sort is descending.

   [#27945]

6. Fixed a bug in REMOVE clause of UPDATE statement, when the path to remove evaluates to NULL. In this case, the UPDATE statement would get into an infinite loop.

   [#27812]

7. Fixed a bug that caused importing data into a table whose shard key is different from one of the tables from which the data was exported to cause the imported data to be corrupted. The data is now imported correctly.

   [#27782] [#27783]

8. Fix a problem that caused the Admin CLI history to not be loaded on startup.

   [#27810]

9. Fixed a problem that could cause the Admin CLI 'plan verify data' command to fail to complete and instead remain in the "RUNNING" state. This problem only occurred in stores with 5 or more shards.

   [#27750]

10. Fixed a problem that could cause the Admin CLI 'plan verify data' command to fail on nodes that had a master change after restarting. The command will now work in those cases, although a master change during the verification of an individual node will still cause that verification to fail and mean that the node's verification needs to be redone. We expect to fix this additional problem in a future release.

    [#27773]

11. Fixed a problem that caused using the stop command to fail when used to stop a kvlite instance. For example:

    ```
    java -jar KVHOME/lib/kvstore.jar stop -root /tmp/mykvlite
    ```

    Failed to stop SNA: Bootstrap config file `./mykvlite/config.xml` does not exist.

    [#27748]

12. Fixed a problem that DDLGenerator did not create correct DDLs for tables with Timestamp or Map of enums.

    [#28010]

13. The 'Shaded' version of the antlr4-runtime 3rd party library that was previously shipped under the name antlr4-runtime.jar is shipped with this release under the name antlr4-runtime-nosql-shaded.jar. Shading that library is necessary to avoid ClassLoader conflicts (typically manifested as a NoClassDefFoundError) when an

application runs in an environment that depends on a different, incompatible version of that library; for example, various containers, Hadoop clusters, Hive clients, etc. When running in those environments, in general, this will require no changes to current applications other than making the corresponding name changes in the application's classpath.

[#27616]

Utility Changes

1. Removed support for the -r2-compat option from the 'table create' command in the Admin CLI. The ability to create tables on top of key/value data was removed in release 19.3 as part of removing support for Avro, but this option had been left in place by mistake.

   [#27907]

Changes in 19.3.12

The following changes were made in Oracle NoSQL Database Release 19.3.12 Enterprise Edition.

Topics

New Features

1. Added support for indexes that do not index `NULL` and non-existent values. This is an index property that can be specified in the index-creation DDL (the default is to index `NULL` and non-existent values). Such indexes may be useful when the data contain a lot of `NULL` and/or non-existent values on the indexed fields, because the time and space overhead of indexing is reduced. However, use of such indexes by queries is restricted. Specifically, they can be used only if for every indexed field, there is a sargable predicate on that field and the predicate in not `IS NULL` or `NOT EXISTS`.

   [#27486]

2. Modified operation and request count statistics to use long values to support stores with larger numbers of operations.

   Added some new JMX bean methods that return total counts as longs. In some cases, for large stores, counts could exceed 2^31, so these new methods are needed to return accurate values for these statistics. The old methods are maintained for compatibility and will return `Integer.MAX_VALUE` if the count gets too large.

   New methods on the `OperationMetrics` interface:

   - `getIntervalTotalOpsLong`
   - `getCumulativeTotalOpsLong`

   New methods on the `RepNodeMXBean` interface:

   - `getIntervalTotalOpsLong`
   - `getCumulativeTotalOpsLong`
   - `getMultiIntervalTotalOpsLong`
   - `getMultiIntervalTotalRequestsLong`
   - `getMultiCumulativeTotalOpsLong`
   - `getMultiCumulativeTotalRequestsLong`

   The new long values are also reported in .perf files, collector .csv files, and in JSON output.

   [#27517]

3. The Streams API now supports dynamically adding or removing subscribed tables to or from any running stream without needing to terminate and recreate the stream.

   [#27381]

4. Users can now use the "describe table" statement to describe the schema of a system table.

   [#27602]

5. Added the new built-in function `regex_like` to the query language. The function is used for string datatype regular expression pattern matching. For example, assume table "persons" has a string field "lastName". The following select statement will qualify records with a lastName starting with the letter "C".

   ```
   select * from persons where regex_like(lastName,"C.*")
   ```

   [#27396]

6. Removed all classes and interfaces from the `oracle.kv.avro` package, which had previously been deprecated. Applications that had been using Avro should be modified to use the table API. *[#27387]*

Bug and Performance Fixes

1. Fix a bug that caused key statistics to be collected more often than the specified collection period.

   [#27615]

2. Allow single-partition queries to use secondary indexes. In previous releases, if a query used a secondary index it would be sent to and executed at all the shards of the queried table. However, if the query also specifies a shard key in the `WHERE` clause, sending it to all the shards is wasteful, because the query will have results only in the partition corresponding to the shard key. To avoid such waste, previous releases would never choose a secondary index if the query specified a shard key (i.e., if it was a single-partition query). Furthermore, if a single-partition contained an `ORDER BY` or `GROUP BY` clause that required the use of a secondary index, the query would raise an error. The current release lifts these restrictions by allowing single-partition queries to use secondary indexes and be executed only in the partition corresponding to the specified shard key.

   [#27573]

3. `NOT EXISTS` predicates were not considered sargable before. Now they are. For example, assume table "persons" has a column "info" of type JSON, and there is an index on info.age. Then, the following query, which looks for persons that do not have any age information, can now use the index on info.age for efficient execution.

   ```
   select * from persons p where not exists p.info.age
   ```

   [#27489]

4. Fixed bug where the `sum()` and `seq_sum()` functions would return 0, instead of `NULL`, if none of the input values were numeric.

   [#27468]

5. Changed the `seq_count()` function to return `NULL` if any of its input items is `NULL`.

[#27582]

6. Fixed a bug where `seq_min` and `seq_max` functions would not skip JSON null if it was the first value in the input sequence.

   [#27628]

7. Changed the way the `EXISTS` and `NOT EXISTS` operators behave when their input expression returns `NULL`. Before, `EXISTS` would return true and `NOT EXISTS` false. Now, they both return `NULL`, unless it is known that the input expression will always return at least one item; in the later case `EXISTS` returns true and `NOT EXISTS` returns false.

   [#27492]

8. Fixed a bug where a value-comparison predicate would be erroneously pushed to a multi-key field of an index.

   [#27491]

9. Fixed two bugs where the GeoJson functions would sometimes throw a `NullPointerException` if an argument is not a valid GeoJson object.

   [#27496] [#27508] [#27526]

10. Fixed a bug where the `geo_distance` function would return -1 instead of `NULL`.

    [#27546]

11. Fixed a bug where SN memory allocation check would sometimes produce misleading logging message.

    [#27524]

12. Modified the `plan migrate-sn` command to continue in the presence of failures if the `-force` flag is specified. Administrators can use this new behavior when migrating SNs from failed zones.

    [#27598]

13. Fix a problem encountered when processing string literals that contain the backslash character.

    [#27640]

14. Fix a bug where using bulk put API to load data to a table with identity type column, the identity column is not filled in value.

    [#27664]

15. Fix a bug where `min()`/`max()`, `seq_min()`/`seq_max()` of Timestamp values with precision less than 9 returns wrong fractional second.

    [#27662]

Utility Changes

1. Removed deprecated Admin CLI commands that supported Avro:

   - `ddl add-schema`
   - `ddl enable-schema`
   - `ddl disable-schema`
   - `table add-schema`

- `put kv -json <schemaName>`
- `get kv -json <schemaName>`
- `aggregate kv`
- `show schemas`

[#27387]

2. Upgraded the `verify-data` plan with the following changes:
   - Made the `verify-data` plan asynchronous to avoid the timeout of RMI requests.
   - Added a feature to provide users a list of corrupt keys associated with the error messages as a part of the plan result.
   - Added two new flags, `-show-corrupt-files` and `-valid-time`. `-valid-time` specifies the amount of time for which an existing verification will be considered valid and not be rerun. The default is '10 minutes'. `-show-corrupt-files` specifies whether to show corrupt JE log files in the plan result. It is disabled by default.

   [#27216]
3. Updated InitialCapacityPlanning.xls (the capacity planning sheet) with the following changes:
   - Misc. cleanups to text and presentation.
   - JVM overheads are now accounted for explcitly when sizing memory requirements.
   - Simplified usage by eliminating explicit use of DbCacheSize.
   - Support for specifying one index along with the table.

   [#27550]
4. The Oracle NoSQL Database Hadoop and Hive integration classes can now be used in environments running CDH6 Hadoop and Hive.
5. A 'shaded' version of the `antlr4-runtime.jar` library is now shipped with this release; where the path prefix of each class specified in that library has been changed from the string `org/antlr` to `oracle/kv/shaded/org/antlr`. This has been done to avoid ClassLoader conflicts (typically manifested as a `NoClassDefFoundError`) when an application runs in an environment that depends on a different, incompatible version of that library; for example, various containers, Hadoop clusters, Hive clients, etc. In general, this will require no changes to current applications. The only case where a change in your application will be necessary is in the unlikely event that the classpath of your application includes the `antlr4-runtime.jar` library shipped in this release of Oracle NoSQL Database, rather than a version of `antlr4-runtime.jar` installed in the application's particular runtime environment. In that case, we urge you to change your application's classpath to reference your own instance of `antlr4-runtime.jar`; as the shaded version of `antlr4-runtime.jar` shipped with this product is considered private to Oracle NoSQL Database.
   [#27616] [#27126]

Changes in 19.1.10

The following changes were made in Oracle NoSQL Database Release 19.1.10 Enterprise Edition.

Topics

Bug and Performance Fixes

1. Fixed the kvclient distribution so that it includes all required jar files.

   [#27684]

Changes in 19.1.8

The following changes were made in Oracle NoSQL Database Release 19.1.8 Enterprise Edition.

Topics

New Features

1. Users can now monitor Admin related metrics using JMX. `AdminMXBean.getEnvMetric` returns a JSON string containing a bundle of environment-related metrics.

   [#27077]

2. Stored Admin events used to be pruned by time period only. Starting this release, they are pruned by the number of events as well. The default maximum number of events is 10,000. The default time period is 30 days.

   [#22640]

3. Enhanced the `CREATE FULLTEXT INDEX` command to now support the creation of text indexes on the contents of a JSON document stored in an Oracle NoSQL Database table. When the elements stored in a column of a given table are JSON documents, the enhancements made to `CREATE FULLTEXT INDEX` allow users to specify which of the document's attributes to index, as well as the data type Elasticsearch should use when indexing the attribute.

   [#27069]

Bug and Performance Fixes

1. Fixed an issue in the implementation of the Streams API where a stream client could get a `oracle.kv.MetadataNotFoundException` during start up, even after the checkpoint table gets created successfully.

   [#27497]

2. Modified store contraction so that the retired shard waits for stream clients to finish up to a configurable time out. A new RN parameter `rnStoreContractWait` is added.

   [#27306]

3. Improved the scalability of metadata propagation for stores with very large numbers of tables.

   [#27257]

4. When creating a secure KVStore handle, an application could get a "Too many open files" exception when `KVStoreFactory.getStore()` was invoked many times. This problem is now fixed by properly closing the Java keystore opens during the SSL initialization.

   [#27322]

5. Fixed a bug that the TableIteratorOption's consistency and requestTimeout configuration were ignored by index iterator and index keys iterator APIs: `tableIterator(IndexKey, MultiRowOptions, TableIteratorOptions)` and `tableKeysIterator(IndexKey, MultiRowOptions, TableIteratorOptions)`.

   [#27345]

6. The number of user plans that were retained in the non-terminal state were previously unbounded. The oldest of these types of plans are now canceled and removed if the number exceeds 1000.

   [#27353]

7. The handling of server security parameters was modified so that the server now enforces a standard set of SSL protocols rather than supporting the protocols enabled in Java. By default, the server now requires SSL to use either the TLSv1.2, TLSv1.1, or TLSv1 protocol. Prior to this change, the server supported any protocol enabled in Java, so clients could be configured to use a non-default protocol without needing to modify the server configuration.

   The `makebootconfig` and `securityconfig` utilities generate configurations that configure clients to use the TLSv1.2, TLSv1.1, and TLSv1 protocols by default. As a result, this change to the default server configuration will only have an effect if the client uses a non-default configuration for SSL protocols. Users can configure their clients to use non-default protocols either by generating a new `client.security` file by using the `makebootconfig` and `securityconfig` utilities, or by modifying the `oracle.kv.ssl.protocols` parameter in either the security file or security parameters. Prior to this change, removing the default setting for this parameter would have permitted using the set of protocols enabled in Java, which includes SSLv2Hello and, in Java 11, TLSv1.3. Now those protocols will only be used if explicitly configured for both clients and servers.

   The `makebootconfig` and `securityconfig` utilities can be used to specify non-default values for this security parameter for a new installation, and `securityconfig` can be used to update the parameter in an existing installation.

   For information on setting security parameters, see Security Configuration.

   Note that there are issues that prevent the use of the TLSv1.3 protocol with Java 11 for secure stores. We expect to enable this new protocol in a future release.

   [#27414]

8. Fixed `TableAPI.getTables("sysdefault")` to return only tables in the sysdefault namespace.

   [#27328]

9. Fixed an exception that occurred when using the NO CACHE value for the GENERATED AS IDENTITY option in a query.

   [#27368]

10. Fixed a problem where the `plan failover` Admin command could produce an incorrectly configured store by leaving electable group size override parameters enabled. The incorrect configuration could cause attempts to restore the original store topology to fail.

    [#27513]

11. Fixed a bug where password contain equal mark cannot be read out from plain text password file store correctly.

[#27525]

12. Fixed a problem where the checkpoint made by Streams API user maybe overridden by an internal checkpoint made during elastic operations, or vice-versa. The incorrect internal checkpoint may cause streams to fail during elastic operations, and user may lose her previously made checkpoint. To support elasticity operations correctly, all stream clients need to be upgraded before performing the elasticity operation.

    [#27521]

13. The import/export utility provides an `overwrite` option that automatically overwrites any record that is found to exist in the Oracle NoSQL Database during import.

    [#27515]

14. Users will no longer get the error "Failed to start SNA: unexpected snapshot operation state" when trying to recover a snapshot taken using a prior release, and then restoring it using a later version of Oracle NoSQL Database. The `snapshot.stat` file was not getting cleaned up. 19.1 users who try to recover snapshots taken using 18.3 will still see this error. You can work around the problem by removing the `RESTORE=STARTED` entry from the `snapshot.stat` file found in the snapshots directory.

    [#27452]

15. Users can restore a backup that had `admindir` specified. Prior to this fix, the Admin failed to come up after a restore from a store that had `admindir` specified. Users using versions prior to 19.1 can do the following command and then restart the SNA without the `-restore-from-snapshot` option:

    ```
    cp -r admindir/snapshots/SNAPSHOT_NAME admindir/recovery
    ```

    [#27454]

16. Oracle NoSQL Database now uses all of the memory it can for the Java heap for heap sizes that are less than 32 GB rather than using some of it for the offheap cache. This change was made to improve performance.

    [#27454]

17. `NullPointerException` no longer occurs when a secure `kvstore` registers with a secure Elasticsearch cluster via the `register-es` Admin CLI command.

    [#27474]

Utility Changes

1. Changed the default consistency policy for Admin CLI and SQL shell to ABSOLUTE.

    [#27350]

2. The Import/Export utility has been enhanced to support JSON and MongoDB JSON formats. The preview migrator utility released in 18.3 is no longer supported.

    [#27422]

Changes in 18.3.11

The following changes were made in Oracle NoSQL Database Release 18.3.11 Enterprise Edition.

Topics

New Features

1. Oracle NoSQL's Enterprise Manager plugin has the new capability to discover store components running on Solaris 10. *[#26836]*

Changes in 18.3.10

The following changes were made in Oracle NoSQL Database Release 18.3.10 Enterprise Edition.

Topics

Utility Changes

1. Added `request-timeout-ms=<milliseconds>` to the config file of export and import utilities to configure the request timeout for iterator operation in export process or bulk put operation in import process. *[#26662]*

Changes in 18.3.9

The following changes were made in Oracle NoSQL Database Release 18.3.9 Enterprise Edition.

Topics

New Features

1. Streams are now supported during elasticity operations such as redistribution, store expansion and contraction, etc. The events applied to a particular key are streamed in the same order that they are applied at the store, regardless of the shard that event was applied to. Prior to this feature an `AdminFaultException` would occur if an elasticity operation was attempted on a store with active subscribers. *[#26662]*
2. Added namespace component to table names.

   Queries use a namespace qualified name of form: `ns:TableName`. DDL adds new statements: `CREATE NAMESPACE` and `DROP NAMESPACE`.

   API changes: `TableAPI.listNamespaces()` returns all known namespaces and `ExecuteOptions.setNamespace()` and `getNamespace()` to specify the namespace of unqualified names used in queries.

   There are 2 new system privileges:

   - `CREATE_ANY_NAMESPACE`
   - `DROP_ANY_NAMESPACE`

   And new privileges with a namespace scope:

   - `CREATE_TABLE_IN_NAMESPACE`
   - `DROP_TABLE_IN_NAMESPACE`
   - `EVOLVE_TABLE_IN_NAMESPACE`
   - `CREATE_INDEX_IN_NAMESPACE`
   - `DROP_INDEX_IN_NAMESPACE`
   - `MODIFY_IN_NAMESPACE`
   - `READ_IN_NAMESPACE`
   - `INSERT_IN_NAMESPACE`
   - `DELETE_IN_NAMESPACE`

   These privileges can be granted or revoked using:

   ```
   GRANT namespace_privilege ON NAMESPACE ns0 TO role_name
   REVOKE namespace_privilege ON NAMESPACE ns0 FROM role_name
   ```

[#26036]

3. Added support for querying GeoJSON data.

> **Note:**
>
> This feature is available only in the Enterprise Edition of Oracle NoSQL Database.

The GeoJSON specification (https://tools.ietf.org/html/rfc7946) defines the structure and content of JSON objects that are supposed to represent geographical shapes on earth (called geometries). Oracle NoSQL Database implements a number of functions that do indeed interpret such JSON objects as geometries and allow for the search for rows containing geometries that satisfy certain conditions. Search is made efficient via the use of special indexes on the GeoJSON data.

As an example consider a table whose rows store points of interest. The table has an id column as its primary key and a poi column of type JSON. Values of the poi column may look like these two JSON documents:

```
{
  "kind" : "city hall",
  "address" : { "state" : "CA",
                "city" : "Campbell",
                "street" : "70 North 1st street"
              },
  "location" : { "type" : "point", "coordinates" : [-121.94, 37.29] }
}

{
  "kind" : "nature park",
  "name" : "castle rock state park",
  "address" : { "state" : "CA",
                "city" : "Los Gatos",
                "street" : "15000 Skyline Blvd"
              },
  "location" : { "type" : "polygon",
                 "coordinates" : [
                                   [
                                     [-122.1301, 37.2330],
                                     [-122.1136, 37.2256],
                                     [-122.0920, 37.2291],
                                     [-122.1020, 37.2347],
                                     [-122.1217, 37.2380],
                                     [-122.1301, 37.2330]
                                   ]
                                 ]
               }
}
```

Both of these documents have a "location" field whose value is a GeoJSON object. In the first document, the GeoJSON represents a single point defined by its coordinates: a

longitude, latitude pair. In the second document, the GeoJSON represents a polygon defined by the coordinates of its vertices.

The following query looks for nature parks in northern California (including parks that straddle the border with neighbor states). The query uses the `geo_intersect` function with a polygon representing northern california as its second argument.

```
select t.poi as park
from PointsOfInterest t
where t.poi.kind = "nature park" and
      geo_intersect(t.poi.location,
                    { "type" : "polygon",
                      "coordinates" : [
                                        [
                                          [-121.94, 36.28],
                                          [-117.52, 37.38],
                                          [-119.99, 39.00],
                                          [-120.00, 41.97],
                                          [-124.21, 41.97],
                                          [-124.39, 40.42],
                                          [-121.94, 36.28]
                                        ]
                                      ]
                    })
```

The following query looks for gas stations within a mile (1609 meters) of a given route including, for each qualifying gas station, its actual distance from the route. The query uses the `geo_near()` function with a `LineString` representing the route as its second argument. Furthermore, the `geo_near` function adds an implicit order-by distance, and as a result, the query orders the returned gas stations by ascending distance from the route.

```
select t.poi as gas_station,
       geo_distance(t.poi.location,
                    { "type" : "LineString",
                      "coordinates" : [
                                        [-121.9447, 37.2975],
                                        [-121.9500, 37.3171],
                                        [-121.9892, 37.3182],
                                        [-122.1554, 37.3882],
                                        [-122.2899, 37.4589],
                                        [-122.4273, 37.6032],
                                        [-122.4304, 37.6267],
                                        [-122.3975, 37.6144]
                                      ]
                    }) as distance
from PointsOfInterest t
where t.poi.kind = "gas station" and
      geo_near(t.poi.location,
               { "type" : "LineString",
                 "coordinates" : [
                                   [-121.9447, 37.2975],
                                   [-121.9500, 37.3171],
                                   [-121.9892, 37.3182],
```

```
                                    [-122.1554, 37.3882],
                                    [-122.2899, 37.4589],
                                    [-122.4273, 37.6032],
                                    [-122.4304, 37.6267],
                                    [-122.3975, 37.6144]
                                   ]
                    },
                    1609)
```

Both of the above queries can be executed efficiently by creating the following index:

```
create index idx_kind_loc on PointsOfInterest(poi.kind as string,
                                               poi.location as geometry)
```

[#27078]

4. Added sequence aggregation functions.

   The following functions were added: `seq_count`, `seq_sum`, `seq_avg`, `seq_min`, and `seq_max`. Contrary to the corresponding SQL aggregate functions (count, sum, etc) the sequence aggregate functions do not imply grouping of rows and do not aggregate values from different rows. Instead, they simply aggregate the items in their input sequence. In doing so, they use the same rules as their corresponding SQL aggregate functions. For example, `seq_sum` will skip any non-numeric items in the input sequence.

   As an example consider a `Users` table with an expenses column that is a map containing the expenses of each user for various categories. Then the following query selects, for each user, their id, the sum of their expenses in all categories except housing, and the maximum of these expenses.

   ```
   select id,
          seq_sum(u.expenses.values($key != "housing")) as sum,
          seq_max(u.expenses.values($key != "housing")) as max
   from Users u
   ```

   [#27082]

5. Implemented SQL INSERT/UPSERT statement.

   This statement is used to construct a new row and insert it in a specified table. If the INSERT keyword is used, the row will be inserted only if it does not exist already. If the UPSERT keyword is used, the row will be inserted if it does not exist already, otherwise the new row will replace the existing one.

   As an example, consider the following CREATE TABLE statement:

   ```
   create table Users (
     id INTEGER,
     firstName STRING,
     lastName STRING,
     otherNames ARRAY(RECORD(first STRING, last STRING)),
     age INTEGER,
     income INTEGER,
     address JSON,
     expenses MAP(INTEGER),
   ```

```
  PRIMARY KEY (id),
)
```

Then the following INSERT statement constructs and inserts a new row into the Users table, if a row with id 10 does not exist already.

```
insert into table users values (
    10,
    "John",
    "Smith",
    [ {"first" : "Johny", "last" : "BeGood"} ],
    22,
    45000,
    { "street" : "Main", "number" : 10, "city" : "Reno", "state" :
"NV"},
    { "travel" : 5000, "books" : 2000 }
) set ttl 3 days
```

If the "upsert" keyword is used in place of "insert" in the above statement, the new row will be inserted if it does not exist already, otherwise it will replace the current version of the row. *[#27006]*

6. It is now possible to cast an integer or long to a timestamp in SQL. The integer/long is interpreted as the number of milliseconds since the epoch. *[#27006]*
7. Added `parse_json` SQL function to convert a string to a JSON instance. *[#27006]*
8. Added support for IDENTITY column for tables.

   Users can create a table with IDENTITY column that uses a sequence generator. IDENTITY column is of numeric datatypes: `INTEGER, LONG, NUMBER` for which the system automatically generates a unique number using an internal sequence generator(SG) that is attached the column. Only one IDENTITY column is allowed for a table. Users can specify the following attributes for the sequence generator: `START WITH (default =1), INCREMENT BY (default=1), MINVALUE (default=minimum value of the datatype), MAXVALUE(default=maximum value of the datatype), CACHE(default=1000), CYCLE or NO CYCLE(default=NO CYCLE).`

   The IDENTITY column can be defined as either GENERATED ALWAYS or GENERATED BY DEFAULT. For the GENERATED ALWAYS option, the system will always generate a value for the IDENTITY column and will return an error if the user specifies a value for it. For the GENERATED BY DEFAULT option, the system will only generate a value for the IDENTITY column if the user did not supply a value.

   Users can also alter the IDENTITY column property and the SG attributes using the ALTER TABLE DDL. New syntax is introduced for defining and modifying the IDENTITY column using the CREATE TABLE and ALTER TABLE syntax respectively. For example:

```
CREATE Table t1 (id INTEGER GENERATED ALWAYS AS IDENTITY
  (START WITH 1 INCREMENT BY 2 MAXVALUE 100),
name STRING, PRIMARY KEY (id))
```

This creates a table `t1` for which the system will generate values 1, 3, 5, ..up to 99. If you want to add a CACHE and CYCLE attribute to the SG, you can issue the following ALTER TABLE statement.

```
ALTER Table t1 (ADD id INTEGER GENERATED ALWAYS AS IDENTITY
  (CACHE 3, CYCLE))
```

If you want to drop the IDENTITY property of the column `id`, users can do so as shown below.

```
ALTER Table t1 (MODIFY id DROP IDENTITY)
```

The system generates the IDENTITY column values during SQL statements - INSERT, UPSERT and UPDATE or Table.Api - PUT, PUTIFPRESENT and PUTIFABSENT. Refer to Oracle NoSQL Database documentation for the semantics for these when an IDENTITY column is involved.

[#25154]

9. Added a new user-specifiable policy parameter for JVM overhead percentage.

   Add a new policy parameter named `jvmOverheadPercent` for the command `change-policy -params [name=value]`. If not specified, the default is `25`.

   [#27089]

10. Added a new Migrator utility. Migrator utility allows user to import MongoDB JSON entries in strict mode representation (exported using mongoexport utility) or normal JSON entries into an Oracle NoSQL Database store. This is a preview version. A general availability version that is integrated with IMPORT/EXPORT will be available in a future release.

11. Added administrative command through REST API. Admin REST API allows user to run admin commands through HTTP or HTTPS requests to Oracle NoSQL Database store. The request and response payload are in JSON format, user can use utility like "curl" to run admin commands against the store.

    [#26596]

12. There is a new configuration option in `KVStoreConfig` that permits specification of a local address on a client machine when connecting to KVStore. Such configuration permits an extra level of network traffic control when running on client machines with multiple NICs. Please review the java doc associated with `oracle.kv.KVStoreConfig.setLocalAddress(InetSocketAddress)` for details.

    [#26879]

Bug and Performance Fixes

1. Fixed a number of casting bugs:
   a. Casting a string to number was wrong.
   b. Any kind of value should be castable to string, but that was not the case.
   c. When casting a map to a record, casting of JSON null was not supported.
   d. Trying to cast an empty string to a timestamp would raise `StringIndexOutOfBoundsException`. Now it raises `IllegalArgumentException` with an appropriate message.

   [#27006]

2. Fixed bug in UPDATE statement: the new version of the row was not set when the statement has a RETURNING clause.

   [#27006]

3. Made the server return the security check failure response to Streams API client.

   [#26091]

4. Fixed a problem where alter table would incorrectly remove a table's Time To Live (TTL) setting.

   [#26983]

5. The following changes to the key statistics parameters were made:

   - The default value for `rnStatisticsSleepWaitDuration` has been changed from `1 second` to `60 seconds`
   - `rnStatisticsGatherInterval` must have a value greater than or equal to `60 seconds`. If `rnStatisticsGatherInterval` is set to a value less than `60 seconds` then 60 seconds is used for its value.
   - `rnStatisticsLeaseDuration` must have a value less than or equal to one day (24 hours). If `rnStatisticsLeaseDuration` is set to a value greater than 1 day the value of 1 day is used.
   - `rnStatisticsSleepWaitDuration` must have a value greater than or equal to `10 seconds` and less than or equal to `rnStatisticsGatherInterval`. If `rnStatisticsSleepWaitDuration` is set to less than `10 seconds` the value of 10 seconds is used. If `rnStatisticsSleepWaitDuration` is set to greater than `rnStatisticsGatherInterval` the value of `rnStatisticsGatherInterval` is used.

   The parameter values are checked on each RN when the RN starts or when the parameter values are changed. If any of the key statistics parameters are overridden one or more warning messages will appear in the RN log.

   [#26939]

6. Corrected the spelling of the `versionCheckInterval` admin service parameter introduced in the 18.1 release. Any values specified for the incorrectly spelled `verionCheckInterval` parameter will be ignored starting with this release and will need to be specified again using the correct spelling.

   [#27000]

7. When upgrading to this release, some elasticity operations, such as redistribute, rebalance and contraction, will require that all nodes be upgraded before they can start or resume.

   The elasticity operations that are affected are those that require data to be migrated from one shard to another.

   If one of these operations is running when the upgrade starts, or is started during the upgrade, the operation will pause until the upgrade is completed. Once the upgrade is complete, the elasticity operation will resume.

   [#26943]

8. The `DurabilityException` class has a new method `getNoSideEffects` which returns whether it is known that the operation that produced this `DurabilityException` had no side effects. Applications that receive a

`DurabilityException` when performing a modify operation and find that `getNoSideEffects()` returns true can safely assume that none of the changes requested by the operation have been performed and then retry the operation. If it returns `false`, then the operation may or may not have had side effects.

[#27073]

9. The memory allocation calculation now takes into account the JVM overhead. Made the `rnHeapPercent` represent the percentage of SN memory reserved for requested Java heap size, and add a new user-specifiable parameter `jvmOverheadPercent`, which represents additional memory used by JVM as a percentage of requested Java heap size. The default value for `jvmOverheadPercent` is `25`.

   The default SN memory allocation is:

   - 85% for Java heap and overhead
     - 68% for requested Java heap size (`rnHeapPercent`)
     - 25% `(jvmOverheadPercent)` * 68 `(rnHeapPercent)` = 17% for JVM overhead
   - 10% for the operating system (`systemPercent`)
   - 5% (the remainder) for the off-heap cache

   [#27089]

10. Fixed a bug where a query executing using `NullValue` external variable won't raise `UnsupportedOperationException`.

    [#27177]

11. There have been several bug fixes in elasticity operations (topology redistribute, rebalance, and contraction) when indexes are present in the store. Included with these fixes is the new ability to rebuild an index is the index becomes corrupted.

    [#26856] [#27024] [#27189]

12. Fixed an issue with exception handling during a query operation which incorrectly caused a RN to restart.

    [#27183]

13. Fixed a bug that could cause an exception such as the following, making the RN unavailable. The bug was present in versions of 18.1 prior to 18.1.20. It can occur under certain conditions when 336 or more tables have been created over the lifetime of the store.

```
2018-08-29 14:16:12.965 UTC SEVERE [rg1-rn1] Process exiting
oracle.kv.impl.rep.EnvironmentFailureRetryException:
com.sleepycat.je.EnvironmentFailureException: (JE +18.1.xx)
Environment must be closed, caused by:
com.sleepycat.je.EnvironmentFailureException:
Environment invalid because of previous exception: (JE 18.1.xx)
rg1-rn1(1):/DATA00/rg1-rn1/env fetchLN of 0x6030/0x95727f6 parent IN=130
IN class=com.sleepycat.je.tree.BIN
lastFullLsn=0x6039/0x1e1138 lastLoggedLsn=0x603a/0x60ac3e
parent.getDirty()=false state=0
expires=2018-09-02.00 LOG_FILE_NOT_FOUND: Log file missing, log is likely
invalid.
Environment is invalid and must be closed.
        at
```

```
oracle.kv.impl.api.RequestHandlerImpl.executeInternal(RequestHandler
Impl.java:934)
        at
oracle.kv.impl.api.RequestHandlerImpl.executeRequest(RequestHandlerI
mpl.java:682)
        at
oracle.kv.impl.api.RequestHandlerImpl.trackExecuteRequest(RequestHan
dlerImpl.java:649)
        at
oracle.kv.impl.api.RequestHandlerImpl.access$100(RequestHandlerImpl.
java:153)
        at
oracle.kv.impl.api.RequestHandlerImpl$2$1.run(RequestHandlerImpl.jav
a:523)
        at
oracle.kv.impl.api.RequestHandlerImpl$2$1.run(RequestHandlerImpl.jav
a:520)
        at
oracle.kv.impl.security.ExecutionContext.runWithContext(ExecutionCon
text.java:192)
...
```

[#27199]

Utility Changes

1. Added `namespace [namespace]` command to Admin CLI and SQL shell to set or clear namespaces for queries and table operations.
2. Added `-namespace <namespaces>` to export and import utilities to export/import all the tables within the specific namespaces.
3. Added `storagedir` and `availableLogSize` information to `verify configuration` and `ping` command output. If RNs hit out-of-disk limit exception then violations are raised in `verify configuration` output. If `availableLogSize` drops below `5 GB` then warning notes are issued in `verify configuration` output.

   [#25458]
4. `Ping` and `verify` will now display read only status for replication nodes.

   [#26469][#26711]

Changes in 18.1.20

The following changes were made in Oracle NoSQL Database Release 18.1.20 Enterprise Edition.

Topics

Bug and Performance Fixes

1. Fixed a bug that could cause an exception such as the following, making the RN unavailable. The bug was introduced in an earlier version of 18.1. It can occur under certain conditions when 336 or more tables have been created over the lifetime of the store.

```
2018-08-29 14:16:12.965 UTC SEVERE [rg1-rn1] Process exiting
oracle.kv.impl.rep.EnvironmentFailureRetryException:
com.sleepycat.je.EnvironmentFailureException: (JE +18.1.xx)
Environment must be closed, caused by:
com.sleepycat.je.EnvironmentFailureException:
Environment invalid because of previous exception: (JE 18.1.xx)
rg1-rn1(1):/DATA00/rg1-rn1/env fetchLN of 0x6030/0x95727f6 parent IN=130
IN class=com.sleepycat.je.tree.BIN
lastFullLsn=0x6039/0x1e1138 lastLoggedLsn=0x603a/0x60ac3e
parent.getDirty()=false state=0
expires=2018-09-02.00 LOG_FILE_NOT_FOUND: Log file missing, log is likely
invalid.
Environment is invalid and must be closed.
        at
oracle.kv.impl.api.RequestHandlerImpl.executeInternal(RequestHandlerImpl.j
ava:934)
        at
oracle.kv.impl.api.RequestHandlerImpl.executeRequest(RequestHandlerImpl.ja
va:682)
        at
oracle.kv.impl.api.RequestHandlerImpl.trackExecuteRequest(RequestHandlerIm
pl.java:649)
        at
oracle.kv.impl.api.RequestHandlerImpl.access$100(RequestHandlerImpl.java:1
53)
        at
oracle.kv.impl.api.RequestHandlerImpl$2$1.run(RequestHandlerImpl.java:523)
        at
oracle.kv.impl.api.RequestHandlerImpl$2$1.run(RequestHandlerImpl.java:520)
        at
oracle.kv.impl.security.ExecutionContext.runWithContext(ExecutionContext.j
ava:192)
...
```

[#27199]

Changes in 18.1.16

The following changes were made in Oracle NoSQL Database Release 18.1.16 Enterprise Edition.

Topics

Bug and Performance Fixes

1. Fixed compatibility bugs which would not allow any queries to be run with a 18.1 server and an older client. *[#26986]*

Changes in 18.1.13

The following changes were made in Oracle NoSQL Database Release 18.1.13 Enterprise Edition.

Topics

New Features

1. Changed the behavior of CLI command `plan stop-service` to ensure store health. After this change, if stopping services using the plan will cause the store to fall into an unhealthy state, the plan will fail with detailed health check information as the output, such as,

   ```
   One of the groups is not healthy enough for the operation: [rg1] Only 1
   primary nodes are running such that a simple majority cannot be formed which
   requires 2 primary nodes. The shard is vulnerable and will not be able to
   elect a new master. Nodes not running: [rg1-rn1]. Nodes to stop: {rg1=[rg1-
   rn2]} ... ...
   ```

   The service can be forcefully stopped by adding the `-force` flag.

   Note that there is one exception. Since `plan stop-service -all-rn` will always result in an unhealthy store, the health check is skipped for such plans and the `-force` flag is not required. *[#22425]*

2. Implemented group by clause and seq_transform expression in SQL for Oracle NoSQL.

   Group-by is similar to the one in the standard SQL. However, in Oracle NoSQL grouping is possible only if there is an index that sorts the rows by the group-by expressions.

   Together with group-by the following aggregate functions were implemented: count(*), count(expr), sum(expr), avg(expr), min(expr), and max(expr).

   The seq_transform expression takes as input a "source" expression and a "mapper" expression. It evaluates the source expr, producing a sequence of zero or more items, and "transforms" this source sequence by evaluating the mapper expr on each item of the source sequence and concatenating the results of these evaluations. The current source item can be accessed by the mapper expr via the $ variable.

   Here is an example that demonstrates both group-by and seq_transform:

   Assume a "sales" table, whose rows look like this:

   ```
   {
     "id":1,
     "sale":
     {
       "acctno" : 349,
       "year" : 2000, "month" : 10, "day" : 23,
       "state" : "CA", "city" : "San Jose", "storeid" : 76,
   ```

```
    "prodcat" : "vegies",

    "items" : [ { "prod" : "tomatoes", "qty" : 3, "price" : 10.0 },
                { "prod" : "carrots",  "qty" : 1, "price" :  5.0 },
                { "prod" : "pepers",   "qty" : 1, "price" : 15.0 }
              ]
  }
}
```

Assume there is the following index on sales:

```
create index on sales (sale.acctno as integer,
                       sale.year as integer,
                       sale.prodcat as string)
```

Then one can write the following query, which returns the total sales per acctno and year: *[#26427]*

```
select t.sale.acctno,
       t.sale.year,
       sum(seq_transform(t.sale.items[], $.price * $.qty)) as sales
from sales t
group by t.sale.acctno, t.sale.year
```

3. Implemented parent-child joins in SQL.

   More generally, this feature allows for joins among tables in the same table hierarchy. Syntactically, this is done by a new NESTED TABLES clause that may appear in the FROM clause of an SQL query. The NESTED TABLES clause specifies a target table and a number of ancestors and/or descendant tables of the target table. This is similar to using a MultiRowOptions object as a parameter to the tableIterator and tableKeysIterator methods of TableAPI. However, NESTED TABLES greatly extends the capabilities of these programatic APIs and provides more standard semantics and better performance as well. Specifically, NESTED TABLES:

   - Is equivalent to a number of LEFT OUTER JOINS and UNION operations, as they are defined by standard SQL.
   - Allows projection: the rest of the query can select any subset of the columns of the participating tables.
   - Allows predicates to be specified on the ancesteros and descendant tables, using the ON clause from standard SQL.
   - Allows the target table to be accessed via a secondary index even when descendant tables are specified.

   As an example, consider an application that tracks a population of users and the emails sent or received by these users. Given that SQL for Oracle NoSQL does not currently support general purpose joins, the emails are stored in a table that is created as a child of users, so that queries can be written that combine information from both tables using the NESTED TABLES clause. The create table statements for the two tables are shown below.

```
create table users(
    uid integer,
```

```
    name string,
    email_address string,
    salary integer,
    address json,
    primary key(id))

create table users.emails(
    eid integer,
    sender_address string,   // sender email address
    receiver_address string, // receiver email address
    time timastamp(3),
    size integer,
    content string,
    primary key(eid))
```

Here are two queries that can be written over the users and emails tables. Given that users.emails is a child table of users, it has an additional column, not included in its create table definition. This implicit column is named uid and has type integer. Normally, its value will be a user id that appears in the users table, but this is not necessary.

```
#
# Count the number of emails sent in 2017 by all users whose salary is
greater
# than 200K
#
select count(eid)
from NESTED TABLES(
       users
       descendants(users.emails ON email_address = sender_address and
                                   year(time) = 2017)
     )
where salary > 200
```

The above NESTED TABLES clause is equivalent to the following left outer join:

```
users u left outer join users.emails e on u.uid = e.uid and
                                          email_address = sender_address
and
                                          year(time) = 2017
#
# For each email whose size is greater than 100KB and was sent by a user
in the
# the users table, return the name and address of that user.
#
select name, address
from NESTED TABLES(users.emails ancestors(users))
where size > 100 and sender_address == email_address
```

The above NESTED TABLES clause is equivalent to the following left outer join: *[#26670]*

```
users.emails e left outer join users u on u.uid = e.uid
```

4. Introduce new JSON output format for admin CLI commands, runadmin CLI command with "-json" flag will display in the new JSON output format. For compatibility, previous admin CLI JSON output are still supported, user can use "-json-v1" to display previous JSON v1 output format. *[#25917]*

5. Introduce a new plan `plan verify-data` that verifies the primary tables and secondary indices for data integrity. The users can run this plan on Admins and/or RepNodes and can choose to verify either the checksum of data records, or the B-tree of databases or both.

   For example:

   ```
   plan verify-data -all-rns
   ```

   verifies both data record integrity and b-tree integrity of primary tables and secondary indices for all RepNodes.

   And:

   ```
   plan verify-data -verify-log disable -verify-btree enable -index
   disable -all-rns
   ```

   verifies the b-tree integrity of primary tables for all RepNodes. *[#26284]*

6. Modify the Admin CLI command to support multiple helper hosts. Now users can use either -helper-hosts or -host/-port to connect to the master admin. The command can find the admin so long as it can contact any services at the given hosts/ports, so the given hosts/ports do not need to have an admin. Two flags -admin-host and -admin-port are removed. Scripts that rely on them can simply remove these two flags as long as the given hosts/ports for -helper-hosts or -host/-port can connect to an SN in the store. *[#26633]*

7. Changed the release numbering convention to use the last two digits of the current year as the major version number, and a sequential number incremented for each release as the minor version number. This numbering scheme matches similar changes being made to other Oracle products, so the Oracle major and minor release numbers (`KVVersion.getOracleMajor()` and `getOracleMinor()`) are now the same as the regular release numbers (`KVVersion.getMajor()` and `getMinor()`). In addition, the release string no longer contains separate Oracle version numbers. Applications should use the `KVVersion` methods to access individual version number fields. If application parse the version string, they will need to be updated to account for the removal of the Oracle version numbers. *[#26756]*

8. Added two new attributes to the RepNodeMXBean that is available via JMX.

   - RepNodeMXBean.getOpMetric returns a JSON string containing a bundle of operation-related metrics.
   - RepNodeMXBean.getEnvMetric returns a JSON string containing a bundle of environment-related metrics.

   These JSON objects have been, and remain, available as the JMX notifications oracle.kv.repnode.opmetric and oracle.kv.repnode.envmetric. They are described in the Run Book. *[#26760]*

9. The Key Distribution Statistics Utility (described in Appendix G of the Admin Guide) has been changed so that if enabled (via

the `rnStatisticsEnabled` parameter) it is scheduled automatically when a RepNode is lightly loaded. As a result, the parameters: `rnStatisticsLowActivePeriod` and `rnStatisticsRequestThreshold`, have been rendered obsolete and are no longer supported. Admin CLI scripts that set these parameters should be updated to eliminate use of these obsolete parameters. *[#26635]*

10. Added a new parameter, `rnStatisticsTTL`, to govern how long data collected by the Key Distribution Statistics Utility remains in system tables once statistics gathering is disabled (via the `rnStatisticsEnabled` parameter) or a table or index has been dropped. The default time-to-live (TTL) is 60 days. The time unit specified must be either days, or hours. *[#26796]*
11. Release version strings now identify which edition is being used. The result of calling `KVVersion.CURRENT_VERSION.toString()` will mention the Client when using the `kvclient.jar` file. For example:

    ```
    18.1.1 2018-01-24 09:06:12 UTC  Build id: 3eef91c0eaf6 Edition: Client
    ```

    If the `kvstore.jar` file is used, then the version string will identify the server edition of the release. *[#24136]*
12. Implemented support for specifying the Admin directory, Admin directory size and RN log directory in the makebootconfig command:

    - `-admindir <directory path>`

      A path to the directory that will contain the environment associated with an Admin Node. In the absence of explicit directory arguments, the environment files are located under the KVROOT/KVSTORE/SN'ID'/Admin'Id'/ directory. This argument is optional in makebootconfig but recommended.

    - `-admindirsize <directory size>`

      The size of the Admin storage directory. This argument is optional in makebootconfig but recommended.

    - `-rnlogdir <directory path>`

      A path to the directory that will contain the log files associated with a Replication Node. For capacity values greater than one, multiple rnlogdir parameters must be specified in makebootconfig, one for each Replication Node that will be hosted on the Storage Node. If rnlogdir is not specified, by default the logs will be placed under the KVROOT/KVSTORE/log directory. This argument is optional in makebootconfig but recommended.

    If `-rnlogdir` is specified, then the je.info, je.config and je.stat files for specific RN will be stored in rnlogdir. In all cases, the je.info, je.stat and je.config files for Admins will be stored under kvroot log directory. *[#26444]*
13. Full Text Search now uses an internal HTTP client which supports HTTPS connections to Elasticsearch.

    FTS does not use the elasticsearch transport client anymore. It now uses it's own HttpClient built over apache's httpasyncclient library. This implies that the port used while registering Elasticsearch cluster changes to http port instead of the transport used in the earlier revision.

In the command shown below:

```
plan register-es -clustername <es_cluster_name> -host <host_name>
                 -port <http_port> -secure true
```

The port specified should be the HTTP port of ES cluster. It was the transport port in the previous release.

As can be seen in the above command, a new secure flag is added whose default value is true. That is, FTS now runs in secure mode by default and this needs some additional certificate set up, as described in the FTS documentation.

If an existing KVStore already has a registered Elasticsearch, then, after the upgrade, it needs to be registered again using the `plan register-es` command. The additional registration is needed because the registered port has changed from the transport port to the HTTP port. *[#26059]*

14. FTS security is only available in Enterprise Edition.

    For the basic and community editions, the -secure flag needs to be set to false explicitly: *[#26781]*

    ```
    plan register-es -clustername <es_cluster_name> -host <host_name>
                     -port <http_port> -secure false
    ```

Bug and Performance Fixes

1. Fixed a bug that could cause temporary failures in the `plan switchover` Admin CLI command. The bug caused plan task `UpdateRepNodeParams` to fail. The failure could be identified by the plan status output, such as,

   ```
   Failures: Task 72 ERROR at 2018-01-01 01:01:01 UTC:
   UpdateRepNodeParams rg1-rn1: 72/UpdateRepNodeParams rg1-rn1 failed.:
   null,
   ```

   together with a message inside the admin log, such as,

   ```
   2018-01-01 01:01:01.001 UTC INFO [admin1] Couldn't update parameters
   for rg1-rn1 because unexpected exception:
   com.sleepycat.je.rep.ReplicaStateException: (JE 7.3.6) (JE 7.3.6)
   GroupService operation can only be performed at master.
   ```
   [#26776]

2. Fixed a bug and improved read availability for when a rep node reaches disk limit. The bug caused a rep node failing to restart after reaching disk limit with a message inside the rep node log, such as,

   ```
   2018-01-01 01:01:01.001 UTC SEVERE [rg1-rn1] Process exiting
   com.sleepycat.je.DiskLimitException: (JE 7.6.3) Disk usage is not
   within je.maxDisk or je.freeDisk limits and write operations are
   prohibited: ... ...
   ```

   The fix also enables the rep node to keep serving read requests after reaching disk limit. *[#26701]*

3. Allow predicates inside path-filtering steps to be used as index-filtering predicates.

   For example, consider the following query:

   ```
   select id
   from Foo f
   ```

```
where exists f.info.address.phones[$element.areacode > 408 and
                                   $element.kind = "home"]
```

Assume we have an index on (info.address.phones.areacode, info.address.phones.kind).

The areacode predicate will be used as a start/stop pred for the index scan. The kind predicate can be used as an index-filtering pred, but before this fix it wouldn't be used as such. If the kind pred is not pushed to the index, we also loose the "covering-index" optimization for this query. *[#26162]*

4. In case of a secondary covering index, locks were being acquired on the table rows, which means that the primary index was being looked up for every such lock. Instead, in this case, only the qualifying index entries need to be locked. This fix can result in significant performance improvement if all data accessed are found in main memory at the server. For example, for a simple prepared query, this fix showed a 20% improvement in the end-to-end query latency. *[#26451]*
5. Avoid data cloning in UPDATE query statements. In previous releases, new values (computed in SET, ADD or PUT clauses) were cloned before they were used to update the target item(s). This is because, it is possible to create cycles among items, resulting in stack overflows when such a circular data structure is serialized. This fix tries to avoid avoid such cloning in most common cases, by detecting at compile time that a cycle is not possible.
6. When a query uses a multi-key index, it is often the case that duplicate results must be eliminated. This is done based on primary key values. A bug was fixed that would cause an exception to be thrown if a primary key column is of type Number or Timestamp. *[#26460]*
7. Fixed a bug that was preventing external variables to be used in update statements. *[#26504]*
8. Fixed a bug that ocurred when the SELECT clause contained a single expression with no AS clause. In this case, the value returned by this expression must be wrapped in a single-field record (because it may not be a record, and queries must always return records having the same record type). However, this wrapping was not always done. *[#26767]*
9. Fixed a query bug showing up when querying a key-only table with an order-by query. A QueryStateEception would be thrown in this case during compilation, because the order-by expressions were not being rewritten to access the index fields instead of the table columns. *[#26632]*
10. Fixed a bug with a select-star query that (a) queries a key-only table and (b) uses a secondary index that indexes all the table columns. The bug would cause the index entries to be returned instead of the table rows. Although, in this case, the index entries contain the same information as the table rows, the ordering of the columns and/or their names may be in different. As part of this fix, an index that indexes all the tale columns will always be recognized as a covering index, whereas this was not the case before. *[#26838]*
11. Fixed a bug showing up in queries like the following example:

```
declare $ext6 string;
select f.str
from foo f
where f.str >= $ext and f.str <= "ab"
```

If the $ext variable is set to "ab", the query will return all rows whose str column starts with "ab" rather than being equal to "ab". Notice that if an external variable were not used, the bug would not show up, because the compiler would convert the WHERE condition to f.str = "ab". *[#26671]*

12. Fixed an issue where the `topology validate` command returns "null" when the topology candidate that is being validated contains Storage Nodes that do not exist in the current store. *[#26294]*
13. There is now a limit on the number of plans stored by the Admin. Plans with an ID 1000 less than the latest plan will be automatically removed from the Admin's persistent store. Only plans that are in a terminal state (SUCCEEDED or CANCELED) are removed. *[#22963]*
14. A limitation has been placed on plan names which prevents the creation of new plans with a name starting with "SYS$". This prefix is reserved to plans which are created internal by the Admin to perform its own administrative operations. Existing plans starting with "SYS$" are not affected. *[#26279]*
15. Fixed some issues that prevented requests from completing within the specified request timeout, in particular when checking for quorum and specifying JE transaction timeouts. Also, modify request handling to prioritize ConsistencyException over RequestTimeoutException, since ConsistencyException provides more specific information about the cause of the request failure. *[#22849]*
16. Fixed a table scan issue where the order of rows returned by the following method on the TableAPI interface may be incorrect when Direction.FORWARD and Direction.REVERSE are specified. The bug required that the primary key field(s) be declared in order at the beginning of the DDL statement to create the table. That is no longer the case. This is the method with the problem:

    `TableIterator<Row> tableIterator(...)` *[#26769]*
17. Made code changes and upgraded some external libraries to support Java 9. Note that these changes are still preliminary: full testing will move to the latest Java version in a future release. *[#25278]*
18. Fixed an issue that Storage Node configured with "-noadmin" flag doesn't suppress starting a Bootstrap Admin. *[#26840]*
19. Fixed an issue that returns "Unknown statement" when execute update statement on sql shell. *[#26357]*
20. Support to parse Timestamp string with zone offset in format of "+HH:MM"/"-HH:MM" or 'Z'(rep for UTC) with default pattern. *[#25808]*
21. Fixed the output message for "show pool -name" command when supplying a non-exist pool name. Previously it will display "Unknown Exception" with stack trace. *[#26639]*
22. Fixed the output of "ping" command to be directed to stdout when the command exit code is 0. Previously all the output of "ping" will be directed to stderr. *[#26693]*
23. Additional optional argument -shard added in ping command to check for services status specific to a particular shard. These changes have been done for both admin and top level ping version. ping -shard shardId will give status for SN, RNs and Arbiter associated with specific shard. Additional information about number of shard in topology added in show topology output with numShard=X. *[#25348]*
24. Fixed a problem which prevented information about storage directories specified with the `-storagedir` and `-storagedirsize` flags to the `makebootconfig`

command from being included in the output of the `generateconfig` command. *[#26353]*

Storage Engine Changes (JE 18.1)

1. Fixed a bug that could on occasion cause a NullPointerException after an RN transitions from Master to Replica with the representative stack trace below: *[#26495]*

```
   2017-08-08 06:59:15.498 UTC SEVERE [rg3-rn1] JE: Uncaught exception in
feeder
   thread Thread[Feeder Output for rg1-rn1,5,main]
nulljava.lang.NullPointerException
    at
com.sleepycat.je.rep.vlsn.VLSNIndex.getLatestAllocatedVal(VLSNIndex.java:4
88)
    at
com.sleepycat.je.rep.impl.node.Feeder$OutputThread.writeAvailableEntries(F
eeder.java:1337)
    at
com.sleepycat.je.rep.impl.node.Feeder$OutputThread.run(Feeder.java:1163)
```

2. JE's per-Database (per-partition) disk utilization metadata is no longer maintained in order to better support very large numbers of partitions. Using a worst case example of 20K partitions per RN with records spread over 1,000 data files, previously the utilization metadata would occupy roughly 2GB of memory and over 100GB of disk. This metadata has been removed. Due to the metadata removal, the dataAdminBytes cache statistic has also been removed. *[#26597]*
3. JE recovery time (RN and Admin startup time after a crash) has been reduced. To go along with this optimization, NoSQL DB has increased the default JE checkpoint interval to reduce writing and improve disk utilization. *[#26179]*
4. Fixed a NullPointerException during deadlock detection on an RN or Admin, for example:

```
java.lang.NullPointerException:
  at
com.sleepycat.je.txn.LockManager$DeadlockChecker.hasCycleInternal(LockMana
ger.java:1942)
...
```

   This was never reported for NoSQL DB, but if it did occur would have caused the RN or Admin to restart. It did not cause persistent corruption. *[#26570]*
5. Fixed two bugs that sometimes caused internal operations to be included in JE operation stats (priSearchOps, etc). *[#26694]*
   - The Btree verifier was incorrectly contributing to operation stats.
   - Deletion operations sometimes incorrectly included search and position stats, depending on the timing.
6. Fixed a very rare, timing related bug that could cause internal corruption. The bug has always been present in JE HA and has been seen only once in a stress test. *[#26706]*
7. The default for JE EnvironmentConfig.TREE_COMPACT_MAX_KEY_LENGTH has been changed from 16 to 42 bytes. This reduces cache usage for Btree internal nodes by 5 to 25%, depending on the key size used (smaller key sizes give larger savings). *[ANDC-203]*

8. Fixed a bug that could cause the following exception on a replica node, when using TTL in conjunction with record deletions:

```
  2018-02-10 12:00:00.006 UTC INFO [rg1-rn1] JE: Replay thread
exiting
  with exception:Environment invalid because of previous
exception: ...
  Replicated operation could not be applied. DEL_LN_TX/14
  vlsn=1,711,027,333 isReplicated="1" txn=-834602813 LOG_INCOMPLETE:
  Transaction logging is incomplete, replica is invalid.
Environment is
  invalid and must be closed. Problem seen replaying entry
DEL_LN_TX/14
  vlsn=1,711,027,333 isReplicated="1" txn=-834602813
```

   The problem can occur under the following conditions:

   - The TTL feature is used.
   - A replica is lagging or down (this is not uncommon).
   - Records with a TTL are also sometimes deleted explicitly at around the time the record expires.

   Prior to this bug fix, the problem could be corrected only by performing a network restore on the replica. *[#26851]*

Utility Changes

1. Snapshot utility will by default create snapshot for configurations as well as service data. Introduced new arguments `-restore-from-snapshot` in "start" command line to allow user directly restore from previous snapshot when starting up SN. *[#26119]*
2. Added new CLI command to limit the type of client requests enabled for the whole store or specific shards.

```
plan enable-requests -request-type {ALL|READONLY|NONE}
     {-shard <shardId[,shardId]*> | -store}
```

   There are three request types can be configured by this command, setting ALL means the store or shards can process both read and write requests; READONLY makes the store or shards only respond to read requests; NONE means no requests will be processed by store or shards. *[#25422]*
3. Release version information provided in the output of the `version` and `ping` commands, and also the `show versions`, `ping`, and `verify configuration` commands in the Admin CLI, now identifies the edition of the release being used. For `version`, the output identifies the edition of the JAR file used for the command. For `show versions`, the server information identifies the edition of the Admin service. For the other commands, the version information for each Storage

Node identifies the edition of the JAR file being used to run that Storage Node. For example: *[#24136]*

```
  java -jar kvstore.jar version
  18.1.1 2018-01-24 09:06:12 UTC  Build id: 3eef91c0eaf6 Edition:
Community
```

And:

```
  java -jar dist/lib/kvstore.jar ping -host localhost -port 5000 \
       -security /tmp/kvroot/security/user.security
  Pinging components of store kvstore based upon topology sequence #14
  10 partitions and 1 storage nodes
  Time: 2018-01-24 21:35:59 UTC   Version: 18.1.1
  Shard Status: healthy:1 writable-degraded:0 read-only:0 offline:0
total:1
  Admin Status: healthy
  Zone [name=KVLite id=zn1 type=PRIMARY allowArbiters=false
      masterAffinity=false]   RN Status: online:1 offline:0
  Storage Node [sn1] on localhost:5000    Zone: [name=KVLite id=zn1
      type=PRIMARY allowArbiters=false masterAffinity=false]
      Status: RUNNING   Ver: 18.1.1 2018-01-24 06:34:44 UTC
      Build id: 3eef91c0eaf6 Edition: Community
    Admin [admin1]        Status: RUNNING,MASTER
    Rep Node [rg1-rn1]    Status: RUNNING,MASTER sequenceNumber:50
            haPort:5006
```

4. Implemented the new feature master affinity zone for KVStore. This feature allows users to set master affinity/no-master affinity for a zone when deploying a zone. The zone with master affinity property has higher priority to host master RNs. Besides, the feature also provides a command to allow users to change the master affinity for the deployed zones. *[#25157]*

   The feature adds new flags **-master-affinity/-no-master-affinity** for the existing command **deploy-zone** and adds a new command **topology change-zone-master-affinity.** The usage of the new command is as follows:

   ```
   Usage: topology change-zone-master-affinity -name <name>
   {-zn <id> | -znname <name>}
   {-master-affinity | -no-master-affinity}
   Modifies the topology to change the master affinity of the specified
   zone.
   ```