Oracle® NoSQL Database

Changelog

Release 21.1
E91819-21
May 2021

Oracle NoSQL Database Changelog, Release 21.1

E91819-21

Contents

Upgrade Requirements

See the Upgrading an Existing Oracle NoSQL Database Deployment section in the *Administrator's Guide*.

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(Abstrac
tResultHandler.java:71)
    at
oracle.kv.impl.async.AbstractResultHandler.checkCallOnResult(Abstrac
tResultHandler.java:49)
    at
oracle.kv.impl.async.AbstractDelegatingResultHandler.onResult(Abstra
ctDelegatingResultHandler.java:49)
    at
oracle.kv.impl.async.AsyncVersionedRemoteDialogInitiator.onCanRead(A
syncVersionedRemoteDialogInitiator.java:157)
    at
oracle.kv.impl.async.dialog.DialogContextImpl$OnCanReadTask.doWork(D
ialogContextImpl.java:1028)
    at
oracle.kv.impl.async.dialog.DialogContextImpl$OnCanReadTask.run(Dial
ogContextImpl.java:1017)
    at java.base/
java.util.concurrent.Executors$RunnableAdapter.call(Executors.java:5
15)
    at java.base/
java.util.concurrent.FutureTask.run(FutureTask.java:264)
    at
oracle.kv.impl.async.dialog.nio.NioChannelExecutor.runTasks(NioChann
elExecutor.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(A
syncVersionedRemoteDialogInitiator.java:131)
    at
oracle.kv.impl.async.dialog.DialogContextImpl$OnCanReadTask.doWork(D
```

```
ialogContextImpl.java:1078)
    at
oracle.kv.impl.async.dialog.DialogContextImpl$OnCanReadTask.run(Dial
ogContextImpl.java:1049)
    at
java.util.concurrent.Executors$RunnableAdapter.call(Executors.java:5
11)
    at java.util.concurrent.FutureTask.run(FutureTask.java:266)
    at
oracle.kv.impl.async.dialog.nio.NioChannelExecutor.runTasks(NioChann
elExecutor.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 20.3.17

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(AbstractCre
atorEndpoint.java:87)
    at
oracle.kv.impl.async.AsyncVersionedRemoteInitiator.startDialog(Async
VersionedRemoteInitiator.java:131)
    at
oracle.kv.impl.async.AsyncVersionedRemoteInitiator.getSerialVersion(
AsyncVersionedRemoteInitiator.java:102)
    at
oracle.kv.impl.async.registry.ServiceRegistryInitiator.getSerialVers
ion(ServiceRegistryInitiator.java:53)
    at
oracle.kv.impl.async.AsyncVersionedRemoteAPI.computeSerialVersion(As
yncVersionedRemoteAPI.java:111)
    at
oracle.kv.impl.async.registry.ServiceRegistryAPI.access$000(ServiceR
egistryAPI.java:47)
    at
oracle.kv.impl.async.registry.ServiceRegistryAPI$Factory.wrap(Servic
eRegistryAPI.java:152)
    at
oracle.kv.impl.util.registry.AsyncRegistryUtils.getRegistry(AsyncReg
istryUtils.java:354)
    at
oracle.kv.impl.util.registry.AsyncRegistryUtils.getRequestHandler(As
yncRegistryUtils.java:314)
    at
oracle.kv.impl.api.rgstate.RepNodeState$AsyncReqHandlerRef.getReques
tHandler(RepNodeState.java:1179)
    at
oracle.kv.impl.api.rgstate.RepNodeState$AsyncReqHandlerRef.resolveIn
ternal(RepNodeState.java:1138)
    at
oracle.kv.impl.api.rgstate.RepNodeState$AsyncReqHandlerRef.resolve(R
epNodeState.java:1083)
    at
oracle.kv.impl.api.rgstate.RepNodeState.resolveReqHandlerRef(RepNode
State.java:251)
    at
oracle.kv.impl.api.rgstate.UpdateThread$ResolveHandler.run(UpdateThr
ead.java:322)
    at java.base/
java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor
.java:1128)
    at java.base/
java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecuto
r.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

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 19.5.19

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

Topics

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

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(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]

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:488)
    at
com.sleepycat.je.rep.impl.node.Feeder$OutputThread.writeAvailableEnt
ries(Feeder.java:1337)
    at
com.sleepycat.je.rep.impl.node.Feeder$OutputThread.run(Feeder.java:1
163)
```

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(Lo
ckManager.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.
```

Changes in 12cR2.4.5.12

The following changes were made in Oracle NoSQL Database 12cR2.4.5.12 Enterprise Edition.

Topics

New Features

1. Introduced Streams API that allows users to subscribe to all the changes from write operations to an Oracle NoSQL Database table. The changes are streamed to your application as a series of discrete `StreamOperation` class objects. These APIs are an implementation of Reactive Streams interface. The Oracle NoSQL version of the streams APIs are prefixed with NoSQL so as to differentiate them from the APIs described by the Reactive Streams standard. For example, Reactive Streams describes a `Publisher` class. Oracle NoSQL Database's implementation of that class is `NoSQLPublisher`. For more information refer to the "Getting Started with Oracle NoSQL Database Streams API". *[#24871]*
2. Added UPDATE statement to perform single-row, server-side updates with SQL.

   The UPDATE statement in Oracle NoSQL Database is an extended version of the one in standard SQL. In addition to the SET clause, which is used to replace the value of existing data fields, the ADD, PUT, and REMOVE clauses have been added to support adding and removing elements and fields to/from arrays and maps. Furthermore, the already powerful path expressions of Oracle NoSQL can be used to identify multiple target fields to update (whether those fields are inside json or strongly typed data) and to compute new values, potentially based on the current values. However, in the current implementation, only a single row may be updated, which implies that the WHERE clause must specify a complete primary key.

   As an example, assume a table, called "People", with only two columns: an integer "id" column and an "info" column of type JSON. Furthermore, let's assume the following row to be updated:

```
{
  "id" : 0,
  "info" :
  {
    "firstName" : "John",
    "lastName" : "Doe",
    "profession" : "software engineer",
    "income" : 200000,
    "address" :
    {
      "city" : "San Fransisco",
      "state" : "CA",
```

```
      "phones" : [ { "areacode":415, "number":2840060,
"kind":"office" },
                   { "areacode":650, "number":3789021,
"kind":"mobile" },
                   { "areacode":415, "number":6096010,
"kind":"home" }
                 ]
    },
    "children":
    {
      "Anna" : { "age" : 10, "school" : "ABC", "friends" : ["John",
"Maria"] },
      "Ron"  : { "age" : 2 },
      "Mary" : { "age" : 7, "school" : "XYZ", "friends" : ["Mark"] }
    }
  }
}
```

The following UPDATE statement updates various fields in the above row:

```
update People p
set p.info.profession = "surfing instructor",
set p.info.address.city = "Santa Cruz",
set p.info.income = $ / 10,
set p.info.children.values().age = $ + 1,
add p.info.address.phones 0 { "areacode":831, "number":5294368,
"kind":"mobile" }
remove p.info.address.phones[$element.kind = "office"]
put p.info.children.Ron { "friends" : ["Julie"] },
where id = 0
returning *
```

The RETURNING clause at the end of the above statement specifies that the whole row, after the update, must be returned. So, the result of the statement looks like this:

```
{
  "id" : 0,
  "info" :
  {
    "firstName" : "John",
    "lastName" : "Doe",
    "profession" : "surfing instructor",
    "income" : 20000,
    "address" :
    {
      "city" : "Santa Cruz",
      "state" : "CA",
      "phones" : [ { "areacode":831, "number":5294368,
"kind":"mobile" },
                   { "areacode":650, "number":3789021,
"kind":"mobile" },
                   { "areacode":415, "number":6096010,
"kind":"home" }
```

```
                ]
    },
    "children":
    {
      "Anna" : { "age" : 11, "school" : "ABC", "friends" : ["John",
"Maria"] },
      "Ron"  : { "age" : 3, "friends" : ["Julie"] },
      "Mary" : { "age" : 8, "school" : "XYZ", "friends" : ["Mark"] }
    }
  }
}
```

The first two SET clauses change the profession and city of John Doe. The third SET reduces his income to one tenth. In this SET, the implicitly declared $ variable is bound to the target item of the SET, i.e., to the result of the p.info.income expression. The fourth SET increases the age of John's children by 1. Notice again the use of the $ variable here: the expression p.info.children.values().age returns 3 ages; the SET iterates over these ages, binds the $ variable to each age in turn, computes the expression $ + 1 for each age, and updates the age with the new value.

The ADD clause adds a new phone at position 0 inside the phones array. The REMOVE removes all the office phones (only one in this example). The PUT clause adds a friend for Ron. In this clause, the expression p.info.children.Ron returns the value associated with the Ron child. This value is a map (the json object { "age" : 3 }) and becomes the target of the update. The 2nd expression in the PUT ({ "friends" : ["Julie"] }) constructs and returns a new map. The fields of this map are added to the target map. *[#26161]*

3. Changes in the cast expression.
   - A record can now be cast to a map. This allows a user to cast a record to a json object.
   - A map can be cast to a record. This is needed for updates, because we don't have a record constructor, so if a user wants to replace a whole record with a new record, the user constructs a map, which is then cast to the record by the update stmt.
   - In the previous implementation casting to JSON was a noop if the input value was a subtype of JSON. However, this was actually a bug, because for example, ARRAY(INT) is a subtype of JSON. and if we want to cast ARRAY(INT) to JSON, we should create an array of ARRAY(JSON) type. So, if the target type is JSON, the current implementation makes sure that any arrays/maps contained in the input value are cast to ARRAY(JSON)/ MAP(JSON) in the output value. Furthermore, since we now allow casting records to maps, casting a record to JSON is now treated as casting the record to MAP(JSON). Before, this would raise an error. *[#26161]*
4. Added SQL function `seq_concat`. It simply returns the concatenation of the sequences returned by its argument expressions. *[#26161]*
5. Added SQL functions to extract temporal fields from a TIMESTAMP field. *[#26046]*
6. Storage directory sizes are now used to enforce disk usage, and new statistics are provided for monitoring disk usage. We strongly recommend that all applications:
   - Specify storage directories and sizes for all RNs.

- Monitor disk usage using the new availableLogSize statistic and take correction action well before this value reaches zero.

Specifying a storage size is important because the storage engine reserves data files for potential replication to nodes that are out of contact. More reserved files are retained for potential replication in this release. If a storage size is not specified, all the free space on the volume (minus 5GB of free space), will eventually be used.

If monitoring is not performed or corrective actions are not taken, there is a danger that the storage size (or the volume size) will be exceeded as the size of the data set grows. Although less likely, this could also occur due to a configuration error (for example, too small a cache size), by neglecting to delete a snapshot, etc. Disk usage is now monitored internally and write operations will be rejected when the storage size (or volume size) is in danger of being exceeded. In this situation, read operations are still allowed. Previously, the RN would cease to function when the volume was filled, i.e., no operations could be performed. Allowing read operations now provides partial availability in this situation. In addition, it is now guaranteed that at least 5GB of free space on the volume will be maintained, regardless of the specified storage size.

The `availableLogSize` statistic represents the amount of space that can be used by write operations, taking into account that reserved data files will be deleted automatically when necessary to perform write operations. Note that monitoring disk usage in the file system is not meaningful, because of the presence of these reserved files that will be deleted automatically.

The `availableLogSize` statistic is one of several new statistics:

- `activeLogSize` -- Bytes used by all active data files: files required for basic operation.
- `reservedLogSize` -- Bytes used by all reserved data files: files that have been cleaned and can be deleted if they are not protected.
- `protectedLogSize` -- Bytes used by all protected data files: the subset of reserved files that are temporarily protected and cannot be deleted.
- `protectedLogSizeMap` -- A breakdown of `protectedLogSize` as a map of protecting entity name to protected size in bytes.
- `availableLogSize` -- Bytes available for write operations when unprotected reserved files are deleted: free space + `reservedLogSize` - `protectedLogSize`.
- `totalLogSize` -- Total bytes used by data files on disk: `activeLogSize` + `reservedLogSize`.

These statistics are included in the `.stat` files and can be monitored using the JMX `oracle.kv.repnode.envmetric` type. In the JMX output, these statistic names are prefixed by "Cleaning_" because they are in the log cleaning (disk garbage collection) statistics group, for example: "Cleaning_availableLogSize".

For a new store, the storage size is specified using the makebootconfig the `-storagedirsize` argument. For an existing store, the storage size can be added (or modified) using the `change-storagedir` plan. Note that in both cases the `-storageDir` must be specified, even if the capacity is one.

Because specifying a storage directory size is recommended, a warning is now printed by makebootconfig when `-storagedir` is specified but `-storagedirsize` is not specified. *[#25220]*

7. `Verify Configuration` now issues a warning if the user does not specify `-storagedirsize` and `-rootdirsize`. We recommend that users set these parameters when installing the store to help manage disk space. *[#26187]*
8. Increased the table name size from 32 characters to 256 characters. *[#26021]*

Bug and Performance Fixes

1. Better index usage by queries.

   In prior releases, conditions that appear inside path filtering steps were not considered as sargable, that is, they would never be used in determining the start or stop points of an index scan. In the current release, this restriction is removed. As an example, consider the following query:

   ```
   select id
   from foo f
   where f.info.address.phones[408 <= $element.areacode and
                               $element.areacode <= 650].kind =any
   "work"
   ```

   If there is an index on `(info.address.phones[].areacode, info.address.phones[].kind)`, all predicates in the query can now be used for the index scan. So, a scan on this index will start at areacode 408 and stop right after areacode 650. During the scan, only entries whose "kind" field is equal to "work" will be selected. Before this fix, if the index was chosen by the query, the full index would be scanned and only the equality pred on kind would applied during this scan.

   In addition to path-filtering predicates, EXISTS predicates are also treated as potentially sargable predicates in this release. For example, consider the following query:

   ```
   select id
   from foo f
   where exists f.info.address.state and exists
   f.info.address.phones.areacode
   ```

   In this case, if we have an index on `(info.address.state, info.address.phones[].areacode)` both predicates in the query are sargable. Two ranges of the index will be scanned. The first range starts at the beginning of the index and finishes at the first entry having EMPTY as the value of the state field. The second range start just after this EMPTY value and finishes at the end of the index. During each scan, entries having the EMPTY value for the areacode field will be eliminated. *[#26044*]
2. Fixed a bug in the case where a query uses a secondary index and it also has a predicate on a primary-key column, which is not part of the index definition. Under some conditions, the predicate on the primary-key column was used as a start/stop predicate for the index scan. This is not possible and the query would raise an `IndexOutOfBoundsException` during compilation. An example is the following:

   ```
   create table t1(id integer, name string, primary key(id));

   create index idx1 on t1(name);
   ```

```
select * from t1 where name = "alex" and id > 0
```

With this bug fix, the condition on the "id" column is not used as a start/stop predicate any more, but it is used as an index-filtering predicate instead. *[#26358]*

3. Added functionality to plan failover command to add, move or remove Arbiters to the topology. *[#25269]*
4. Fixed performance bug in SQL query processing. Removed an unnecessary remote call to fetch table metadata. This call was adding a one-time overhead of close to 2ms to each SQL query. For queries that access few rows, this is a huge overhead, especially when the rows are cached in memory. For example, the latency of a query that accesses and returns a single memory-cached row was 5x slower than an equivalent table iterator API call. *[#26232]*
5. Fixed a query bug that occurs when a TableQuery operation is forwarded by a server to another server. For example, this may happen during partition migration. The bug was in the serialization method of TableQuery, because that method assumed that serialization occurs only when the TableQuery is sent from a client to the servers. *[#26385]*
6. Users can now set the `PASSWORD LIFETIME` to zero. The password will never expire if the value is set to zero days. Prior to this fix, when the user tried to issue "ALTER USER admin PASSWORD LIFETIME 0 DAYS", the statement will fail with message "Time value must not be zero or negative". *[#26040]*
7. Changed the way the Java driver configures Java loggers to make it easier to configure additional log handlers. Applications can now be configured to send logging output to a file by specifying a file logging handler for the "kv.oracle" logger. *[#26134]*
8. Fixed bugs that allow queries to execute against a secure kvstore after a session timeout without returning `AuthenticationRequiredException`. *[#26250], [#26249]*
9. The method `oracle.kv.table.Index.createIndexKey(RecordValue value)` has been deprecated in this release, this method is not able to construct index keys for *multi-key* indexes that include elements of a map or array and can result in multiple index entries, or distinct IndexKey values for a single row. *[#26211]*
10. Fixed a bug where altering a key-only table by adding a non-primary-key column would cause subsequent queries and index creation to fail. This bug exists in R4.4 as well. *[#26354]*
11. Fixed the output for `SHOW AS JSON USERS` and `SHOW AS JSON ROLE role` to output results in the correct JSON format. The incorrect output format was causing parsing issues in their applications. *[#26355]*
12. When upgrading from a pre-4.2 release to 4.2, 4.3, or 4.4, the presence of multiple outstanding plans in the ERROR state can cause the Admin to fail when starting up. That bug has been fixed in this release. *[#26303]*

Utility Changes

1. Introduced two new arguments `-registry-open-timeout` and `-registry-read-timeout` for Admin CLI and ping utility, which are used to configure open and read timeout in milliseconds associated with the sockets used to make registry requests. *[#24164]*

Storage Engine Changes (JE 7.5)

1. Fixed a compatibility problem with the Azul Zulu JVM. Previously the following exception would occur when using Zulu: *[#26163]*

   ```
   The database environment could not be opened:
   java.lang.IllegalStateException:
   Could not access Zing management bean. Make sure -
   XX:+UseZingMXBeans was
   specified.
   ```

2. Fixed a bug that could cause OutOfMemoryError when performing a network restore. This could cause an RN to unnecessarily restart and retry the network restore. (A network restore is used when an RN has been down for some period of time, and is lagging the master node in its shard.) *[#26305]*

3. Fixed a bug that could prevent performing a network restore, after a prior network restore was aborted or incomplete for any reason. (A network restore is used when an RN has been down for some period of time, and is lagging the master node in its shard.)

   For example, this could occur if the RN process is killed during the first network restore, and then another network restore is attempted. The problem could occur only in an environment with a relatively large data set, specifically where at least one billion write transactions had been performed. An example stack trace is below.

   ```
   java.lang.NumberFormatException: For input string: "7473413648"
   at java.lang.NumberFormatException.forInputString(
           NumberFormatException.java:65)
   at java.lang.Integer.parseInt(Integer.java:583)
   at java.lang.Integer.parseInt(Integer.java:615)
   at com.sleepycat.je.rep.InsufficientLogException.init(
           InsufficientLogException.java:218)
   at com.sleepycat.je.rep.impl.RepImpl.handleRestoreRequired(
           RepImpl.java:2296)
   at com.sleepycat.je.recovery.RecoveryManager.findEndOfLog(
           RecoveryManager.java:543)
   at com.sleepycat.je.recovery.RecoveryManager.recover(
           RecoveryManager.java:339)
   at com.sleepycat.je.dbi.EnvironmentImpl.finishInit(
           EnvironmentImpl.java:841)
   at com.sleepycat.je.dbi.DbEnvPool.getEnvironment(DbEnvPool.java:222)
   at
   com.sleepycat.je.Environment.makeEnvironmentImpl(Environment.java:26
   7)
   at com.sleepycat.je.Environment.init(Environment.java:252)
   at com.sleepycat.je.rep.ReplicatedEnvironment.init(
           ReplicatedEnvironment.java:607)
   at com.sleepycat.je.rep.ReplicatedEnvironment.init(
           ReplicatedEnvironment.java:466)
   at oracle.kv.impl.rep.RepEnvHandleManager.openEnv(
           RepEnvHandleManager.java:628)
   at oracle.kv.impl.rep.RepEnvHandleManager.renewRepEnv(
   ```

```
        RepEnvHandleManager.java:465)
at oracle.kv.impl.rep.RepNode.startup(RepNode.java:913)
```

This has been fixed. Without the fix, a workaround for the problem is to remove all the .jdb files on the RN node, and restart the RN. *[#26311]*

Changes in 12cR2.4.4.6

The following changes were made in Oracle NoSQL Database 12cR2.4.4.6.

Topics

New Features

1. Added indexing for JSON data.

   As of the current release, SQL for Oracle NoSQL supports typed JSON indexes. As their name implies, such indexes place some type-related restrictions on the JSON data that is being indexed. Creation of a typed JSON index will fail if the associated table contains any rows with data that violate the restrictions imposed by the index. Similarly, an insert/update operation will be rejected if the new row does not conform to the restrictions imposed by one or more existing JSON indexes. However, as long as the type constraints are satisfied, typed JSON indexes are very similar to indexes on strongly typed data, both in the way their contents are evaluated as well as in the ways they are used by queries.

   Syntactically, the only difference between JSON and non-JSON indexes is that a type must be specified in the CREATE INDEX statement for each index path that indexes a JSON field. The type must be one of the JSON atomic types (a numeric type, or string, or boolean). Such a type declaration implies that the items returned by the index path expression, when evaluated on a table row, must be of the specified type, or SQL NULL, or JSON null. It is also allowed for the indexed path to not exist in the data; in this case the path expression returns an empty result and a special (internal) value EMPTY is placed in the index.

   Like non-JSON indexes, JSON indexes may be "simple" or "multikey". The later are used to index all the elements of an array, or all the entries of a JSON object. An example for each kind of index is shown below. They examples use the following table:

   ```
   create table foo (id integer,
                     firstName string,
                     lastName string,
                     address json,
                     primary key(id));
   ```

   A sample row of the above table may look like this (shown in JSON format):

   ```
   {
     "id":0,
     "firstName" : "John",
     "lastName" : "Doe",
     "address": {
       "street" : "somewhere",
       "city": "San Francisco",
   ```

```
    "state"  : "CA",
    "phones" : [ { "area":408,  "number":5039567, "kind":"work" },
                 { "area":415,  "number":2854026, "kind":"work" },
                 { "area":null, "number":8390129, "kind":"home" },
               ]
  }
}

create index idx1 on foo (address.state as string,
                          address.city as string)
```

In the above index, both the `address.state` and `address.city` paths must return strings, or NULL, or JSON null, or EMPTY (an SQL NULL may be returned only if the top-level `address` column is NULL). Furthermore, since the paths do not contain any multikey steps ([], or .values(), or .keys()), they must return at most one item, which implies that none of the fields in the paths (`address`, or `address.state`, or `address.city`) may be arrays.

```
create index idx2 on foo (address.state as string,
                          address.phones[].area as integer,
                          address.phones[].kind as string)
```

In the above index, the paths `address.phones[].area` and `address.phones[].kind` are multikey, because they contain the [] step. This implies that `address.phones` is expected to be an array (and none of the other fields in these 2 paths can be arrays). However, it is also allowed for `address.phones` to be a single JSON object, or even an atomic value. In general, the index paths are evaluated the same way as path expressions in DML queries. So, if `address.phones` is a single object having area and kind fields, the values of these fields will be indexed. If `address.phones` is an atomic value, the result of the path expression is empty, and EMPTY will be placed in the index. *[#25509]*

2. An EMPTY value is used internally to represent cases where an expression returns an empty result. Applications do not normally have to deal with EMPTY values. The only exception is in IndexKey instances: when a TableIterator is used to scan an index and return index keys, the EMPTY value may appear in the returned IndexKey instances. The `FieldValue.isEMPTY()` method can be used to check if the value of an IndexKey field is EMPTY. Applications may also use the `IndexKey.putEMPTY()` method to search an index for entries containing the EMPTY value in one or more of their fields. *[#25509]*
3. Added NUMBER data type. The NUMBER data type represents the `java.math.BigDecimal` value. It can be used as a universal numeric type that can handle numeric values of any size and precision. A field of type NUMBER can be used as a field of a primary key or an index key. All query expression that work on numbers have been extended to handle NUMBER values as well. *[#25447]*
4. Added operators IS NULL and IS NOT NULL in SQL for NoSQL. As in standard SQL, the operators test whether the result of an expression is (or is not) the SQL NULL. The operators are "sargable", that is, they can be evaluated as conditions on index keys, if appropriate indexes exist. *[#25809]*
5. Indexing of NULL values was introduced in 4.2. However, in 4.2 and 4.3 NULL was also used to index "missing" values, or more precisely, in cases where the

evaluation of an index path on a row would return an empty result (EMPTY). In the current release, NULL and EMPTY are indexed separately. This is required to make the IS (NOT) NULL operators sargable. To see why, consider the following example:

```
create table foo (id integer, map MAP(RECORD(f1 integer, f2
integer)))

create index idx on foo (map.somekey.f1)

select * from foo where map.somekey.f1 IS NULL

Row R1: { "id" : 1, "map" : null }

Row R2 : { "id : 2, "map" : { } }

Row R3 : { "id : 3, "map" : { "somekey" : {"f2" : 10} } }
```

Rows R1 and R3 are in the result set of the query, but not R2 (because on R2, map.key.f1 returns EMPTY).

In 4.2, the index will contain NULL for all of the above rows. In the current release, it will contain NULL for R1 and R3, and EMPTY for R2. So, in the current release, the index can be used to optimize the query. *[#25509]*

6. Allow quoted strings as steps in index paths. This is needed to be able to index a specific entry of a map, when the key of that entry in not a simple identifier. For example:

```
CREATE TABLE Foo(id INTEGER, map MAP(INTEGER)), primary key(id))

create index idx on Foo (map."@abc")
```

Prior to this release, creating the above index was not possible, because quoted strings were not allowed as steps. An attempt to use the path `map.@abc` would fail, because `@abc` is not a valid identifier (due to the '@' char). *[#25854], [#25958], [#25963], [#25974]*

7. Any escape sequences that appear in string literals inside a query are converted (inlined) to their corresponding unicode character when the query is parsed. This is required, because the JSON data model applies the same conversion. So, for example, when JSON text is loaded into Oracle NoSQL, any escape sequences in the JSON text are inlined. As a result, the string literal appearing in queries must have the same "format" internally, as the one used by the stored data. *[#25767]*
8. Added a new plan `plan network-restore` to perform network restore from one RepNode to another within the same shard. This plan can be used to restore data from a secondary node to a primary RepNode; restore a replica from another one if the shard lose quorum and no master available and also could be used to restore a RepNode having corrupted data. *[#25834]*
9. Since this release, configuration files of storage node, Admin, RepNode and security are set to owner read only by default. For example, the "config.xml" in root directory and "security.xml" in security configuration directory. *[#25835]*
10. Oracle NoSQL Database store passwords of users after hashing in the security metadata database, the hashing algorithm was using `PBKDF2WithHmacSHA1`. Since

this release, replace with stronger algorithm `PBKDF2WithHmacSHA256`. Note that this change won't update hashes of existing user passwords. It's recommended to update user passwords to make use of the new algorithm after upgrade. *[#26014]*

11. Supports the "describe table" and "describe index" output in tabular format. *[#25720]*
12. Displays the "describe table" and "describe index" output in tabular format. *[#25720]*
13. The progress of a plan will be available via JMX. A JMX notification is generated to report on the status of a plan and its associated tasks for both serial and parallel plans. It reports on both the general tasks as well as tasks associated with elasticity operations such as migrate partition as shown in the sample out below. *[#25200]*

```
{
  "planId": 17,
  "planName": "Deploy Topo (17)",
  "reportTime": 1483445963460,
  "state": "RUNNING",
  "attemptNumber": 1,
  "migratePartition_Total": 150,
  "generalTask_Total": 14,
  "migratePartition_Successful": 15,
  "generalTask_Successful": 13,
  "migratePartition_Running": 135,
  "generalTask_NotStarted": 1
}
```

Bug and Performance Fixes

1. In previous releases, the "covering index" optimization was not being considered for queries that did not have a WHERE clause. This has been fixed in the current release. As a result, a query like `select id1 from foo`, where "id1" is a primary key column of table "foo", will be evaluated by a keys-only scan over the primary index of the table, instead of retrieving all the table rows. *[#25509]*
2. A bug has been fixed for order-by queries where the order-by clause includes primary-key columns. Because sorting is indexed-based, **all** of the indexed fields must appear in the order-by clause, before the primary-key columns. Otherwise, the query should be rejected. This restriction was not enforced in prior releases, which could lead to wrongly sorted results. *[#25509]*
3. Certain combinations of queries and indexes on maps would cause an error during query execution. The most important case is when an index indexes more than 2 specific keys of a map and a query is using the index to evaluate one or more of its expressions during the index scan using only the index fields (without the need to retrieve the full table row). Here is an example that demonstrates such a case:

```
CREATE TABLE Foo(
   id INTEGER,
   g  LONG,
   rec RECORD(a INTEGER,
              b ARRAY(INTEGER),
              c MAP(RECORD(ca INTEGER, cb INTEGER, cc INTEGER, cd
INTEGER))),
```

```
primary key(id))

create index idx on Foo (rec.c.key1.ca, rec.c.key2.ca,
rec.c.key3.cb)
```

Query:

```
select id from Foo f where f.rec.c.key1.ca >= 3 and f.rec.c.key2.ca
= 20
```

In this query, the 1st predicate in the WHERE clause is used as the start condition for the index scan. This does not cause any problems. The problem is caused by the second predicate, which is used as a "filtering" predicate: it is evaluated as the index is being scanned, using the 2nd field ("rec.c.key2.ca") of the current index entry. Furthermore, the index indexes 3 specific map keys ("key1", "key2", and "key3"). If the path "rec.c.key3.cb" was not part of the index definition, the bug would not show up. *[#25822]*

4. The fact that map keys are case-sensitive was not taken into account when the query processor would consider using an index on a specific map key. This could lead to the wrong index being used by a query. For example:

```
CREATE TABLE Foo (id INTEGER, map MAP(INTEGER), primary key(id))

CREATE INDEX idx ON Foo (map.SomeKey)

select id
from Foo f
where f.map.somekey = 3
```

The above query would use the index. But this would be wrong because the map keys "SomeKey" and "somekey" are not the same.

The case-sensitivity of the map keys was also not taken into account when, during and INDEX CREATE statement, a check is made to see if an index with the same fields exists already. *[#25959]*

5. Elimination of duplicate query results was not done in some cases. Specifically, if a query used a multikey index and no predicates were pushed to that index, duplicate elimination was not done. An index will be used by a query, even if no predicates are pushed into ti, if the index is used for sorting (order-by queries) or if its use is forced by an index hint. *[#26008]*

6. Fixed several bugs in comparison operators. The bugs can cause a query to return wrong results or raise an NPE. Fortunately the bugs show up in corner cases only. *[#25832]*

   - A bug can occur when long or double literal is compared with an expression whose type is integer or float, respectively, and the long/double literal is outside the range of integers/floats. Furthermore, the expression must return one or more NULLs, or an empty result, or more than one items and the operator is a value comparison operator.
   - A bug can occur when long or double literal is compared via != or !=any with an expression whose type is integer or float, respectively, and the long/double literal is outside the range of integers/floats.

- A bug occurs when JSON null is compared with an expression whose type does not contain JSON null, and either (a) the expression returns SQL NULL, or (b) the expression returns EMPTY (the empty sequence) and the comparison operator is !=any.
- A bug occurs when a multi-valued path expression is compared with a const, the path expression matched a multi-key path of an index, the query will use that index, and the comparison predicate is converted to a start/stop condition for the index scan. For example, `t.array[] = 5` where "array" is a column of table t with type array(integer). Normally, this predicate will raise an error if there is any row where the array contains more than 1 elements. However, if there is also an index on "array" and the predicate is pushed to that index, the error could be lost, and the query would return a result.

7. Fixed a compilation bug in array filter expression, when the input expression has EMPTY type. For example, the following query will raise an `EmptyStackException: select f.info.children.keys().age[$element > 4] from foo f` In the path expression appearing in the SELECT clause, the keys() step will always return a bunch of strings. As a result, the .age step on those strings will always an empty result. So, the compiler would assign the type EMPTY to the `f.info.children.keys().age` expression. This was not handled correctly by the following array-filter step ([$element > 4]). *[#25877]*
8. The code that parses JSON text and maps it to a strongly-typed table schema was not handling invalid JSON documents correctly. This has been fixed in this release. *[#25842]*
9. Clarified in the documentation that the values returned by toByteArray (and related) methods are not guaranteed to work with older releases. Byte array values created with either the current or earlier releases can passed to the associated methods that accept byte array values for the current or later releases, but values created by later releases are not guaranteed to be compatible when passed to methods for earlier releases.

   These are the affected methods:

   - `Consistency.toByteArray` and `fromByteArray`
   - `Durability.toByteArray` and `fromByteArray`
   - `ExecutionFuture.toByteArray` and `KVStore.getFuture`
   - `Key.toByteArray` and `fromByteArray`
   - `KeyRange.toByteArray` and `fromByteArray`
   - `Value.toByteArray` and `fromByteArray`
   - `Version.toByteArray` and `fromByteArray`
10. Fixed the RMI registry filter issue occurs while running a secure store with Java SE Development Kit 8, Update 121 (JDK 8u121), which is caused by the Java new feature called RMI better constraint checking. The fix is to add patterns `oracle.kv.**;java.lang.Enum` to RMI registry filter `sun.rmi.registry.registryFilter` automatically if they are not present while starting a Storage Node Agent for both secure and non-secure stores. *[#25923]*
11. Changed the SSL cipher suites preference order to favor suites using GCM cipher algorithm mode if no user-specified cipher suite is configured. In our experimental performance test, it is tracked that suites using GCM provide better performance, particularly on hardware with limited sources of secure randomness. *[#25949]*

12. Changed the error message when issue a query on a closed store handle to be more understandable. *[#25883]*
13. Fixed the issue that failover operation may fail if a secondary zone has more recent data than existing alive primary zones. The fix added potential data loss verification to detect if there is a data loss after failover. To retain data in secondary zones, users need to perform network restore first from leading secondary nodes to primaries. If not, specify force and re-execute the failover plan. *[#24772]*
14. Fixed the issue that the '0.0' in JSON string is failed to be parsed as a float value. *[#25983]*
15. Fixed the issue that restarting a secured KVLite in EE version may fail with IllegalStateException "Unable to access the configured wallet store". This is due to concurrent attempts to open the wallet file. *[#25990]*
16. Upgraded the Elasticsearch library to v2.4.4. *[#25943]*
17. Fixed the issue where the Admin failed to restart on a IllegalCommandException. When an Admin node restarts and becomes a master, it restarts all the plans that were running at the time of failover. Prior to this fix If IllegalCommandException occurred during the restarting of a plan, the Admin may fail to restart. *[#26022]*
18. Fixed the issue for bulk put API where the entries with duplicate key supplied by a single stream are marked as existed before the key is actually found in store. *[#25903]*

Utility Changes

1. Change field name from "datetime" to "reportTime" in following type of SN JMX notification JSON format data. *[#25979]*
   - `oracle.kv.repnode.opmetric`
   - `oracle.kv.repnode.envmetric`
   - `oracle.kv.repnode.replicationstate`
   - `oracle.kv.repnode.status`
   - `oracle.kv.plan.status`
2. Added "put" command to sql shell to put row(s) into the named table: `put -table <name> [-json <string> | -file <file> [JSON | CSV]]`

Changes in 12cR1.4.3.11

The following changes were made in Oracle NoSQL Database 12cR1.4.3.11.

Topics

- Bug and Performance Fixes

Bug and Performance Fixes

1. Updates to improve security related to JLine and the shell command line history. *[#25940]*
2. Fixed an RMI registry filter failure that could occur when running the Oracle NoSQL Database in a secure configuration using Java SE Development Kit 8, Update 121 (JDK 8u121). The failure was caused by the new *RMI better constraint checking* feature. The fix adds the patterns `oracle.kv.**;java.lang.Enum` to the RMI registry filter `sun.rmi.registry.registryFilter` automatically if they are not present when starting a Storage Node Agent. *[#25923]*

   The failures would produce stack traces like the following:

```
Jan 25, 2017 11:15:45 AM java.io.ObjectInputStream filterCheck
INFO: ObjectInputFilter REJECTED: class java.lang.Enum, array
length: -1, nRefs: 14, depth: 4, bytes: 768, ex: n/a
KVLite: exception in start: java.rmi.RemoteException: Can't rebind
snaService at localhost:5000 csf: <SSLClientSocketFactory name=$|
sna|main id=1690670528 connectMs=0 readMs=0 kvStoreName=null
clientUse=USER> ssf: <SSLServerSocketFactory backlog=1024 port
range=0,0 ssl control =
oracle.kv.impl.security.ssl.SSLControl@5836ad63>; nested exception
is:
    java.rmi.ServerException: RemoteException occurred in server
thread; nested exception is:
    java.rmi.UnmarshalException: error unmarshalling arguments;
nested exception is:
    java.io.InvalidClassException: filter status: REJECTED
    at
oracle.kv.impl.util.registry.RegistryUtils.rebind(RegistryUtils.java
:853)
    at
oracle.kv.impl.sna.StorageNodeAgent.bindUnregisteredSNA(StorageNodeA
gent.java:974)
    at
oracle.kv.impl.sna.StorageNodeAgent.startupUnregistered(StorageNodeA
gent.java:793)
    at
oracle.kv.impl.sna.StorageNodeAgent.start(StorageNodeAgent.java:624)
    at
oracle.kv.impl.sna.StorageNodeAgentImpl.start(StorageNodeAgentImpl.j
ava:133)
    at oracle.kv.util.kvlite.KVLite.startSNA(KVLite.java:298)
    at oracle.kv.util.kvlite.KVLite.start(KVLite.java:524)
    at oracle.kv.util.kvlite.KVLite.start(KVLite.java:513)
```

```
    at oracle.kv.util.kvlite.KVLite.main(KVLite.java:647)
    at oracle.kv.impl.util.KVStoreMain$1.run(KVStoreMain.java:190)
    at oracle.kv.impl.util.KVStoreMain.main(KVStoreMain.java:477)
Caused by: java.rmi.ServerException: RemoteException occurred in
server thread; nested exception is:
    java.rmi.UnmarshalException: error unmarshalling arguments;
nested exception is:
    java.io.InvalidClassException: filter status: REJECTED
    at
sun.rmi.server.UnicastServerRef.oldDispatch(UnicastServerRef.java:46
0)
    at
sun.rmi.server.UnicastServerRef.dispatch(UnicastServerRef.java:294)
    at sun.rmi.transport.Transport$1.run(Transport.java:200)
    at sun.rmi.transport.Transport$1.run(Transport.java:197)
    at java.security.AccessController.doPrivileged(Native Method)
    at sun.rmi.transport.Transport.serviceCall(Transport.java:196)
    at
sun.rmi.transport.tcp.TCPTransport.handleMessages(TCPTransport.java:
568)
    at
sun.rmi.transport.tcp.TCPTransport$ConnectionHandler.run0(TCPTranspo
rt.java:826)
    at
sun.rmi.transport.tcp.TCPTransport$ConnectionHandler.lambda$run$0(TC
PTransport.java:683)
    at java.security.AccessController.doPrivileged(Native Method)
    at
sun.rmi.transport.tcp.TCPTransport$ConnectionHandler.run(TCPTranspor
t.java:682)
    at
java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor
.java:1142)
    at
java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecuto
r.java:617)
    at java.lang.Thread.run(Thread.java:745)
    at
sun.rmi.transport.StreamRemoteCall.exceptionReceivedFromServer(Strea
mRemoteCall.java:276)
    at
sun.rmi.transport.StreamRemoteCall.executeCall(StreamRemoteCall.java
:253)
    at sun.rmi.server.UnicastRef.invoke(UnicastRef.java:379)
    at sun.rmi.registry.RegistryImpl_Stub.rebind(Unknown Source)
    at
oracle.kv.impl.util.registry.RegistryUtils$ExceptionWrappingRegistry
.rebind(RegistryUtils.java:1339)
    at
oracle.kv.impl.util.registry.RegistryUtils.exportAndRebind(RegistryU
tils.java:1184)
    at
oracle.kv.impl.util.registry.RegistryUtils.rebind(RegistryUtils.java
:849)
    ... 10 more
```

```
Caused by: java.rmi.UnmarshalException: error unmarshalling
arguments; nested exception is:
    java.io.InvalidClassException: filter status: REJECTED
    at sun.rmi.registry.RegistryImpl_Skel.dispatch(Unknown Source)
    at
sun.rmi.server.UnicastServerRef.oldDispatch(UnicastServerRef.java:45
0)
    at
sun.rmi.server.UnicastServerRef.dispatch(UnicastServerRef.java:294)
    at sun.rmi.transport.Transport$1.run(Transport.java:200)
    at sun.rmi.transport.Transport$1.run(Transport.java:197)
    at java.security.AccessController.doPrivileged(Native Method)
    at sun.rmi.transport.Transport.serviceCall(Transport.java:196)
    at
sun.rmi.transport.tcp.TCPTransport.handleMessages(TCPTransport.java:
568)
    at
sun.rmi.transport.tcp.TCPTransport$ConnectionHandler.run0(TCPTranspo
rt.java:826)
    at
sun.rmi.transport.tcp.TCPTransport$ConnectionHandler.lambda$run$0(TC
PTransport.java:683)
    at java.security.AccessController.doPrivileged(Native Method)
    at
sun.rmi.transport.tcp.TCPTransport$ConnectionHandler.run(TCPTranspor
t.java:682)
    at
java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor
.java:1142)
    at
java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecuto
r.java:617)
    at java.lang.Thread.run(Thread.java:745)
Caused by: java.io.InvalidClassException: filter status: REJECTED
    at
java.io.ObjectInputStream.filterCheck(ObjectInputStream.java:1244)
    at
java.io.ObjectInputStream.readNonProxyDesc(ObjectInputStream.java:18
32)
    at
java.io.ObjectInputStream.readClassDesc(ObjectInputStream.java:1713)
    at
java.io.ObjectInputStream.readNonProxyDesc(ObjectInputStream.java:18
29)
    at
java.io.ObjectInputStream.readClassDesc(ObjectInputStream.java:1713)
    at
java.io.ObjectInputStream.readEnum(ObjectInputStream.java:1938)
    at
java.io.ObjectInputStream.readObject0(ObjectInputStream.java:1532)
    at
java.io.ObjectInputStream.defaultReadFields(ObjectInputStream.java:2
231)
    at
java.io.ObjectInputStream.readSerialData(ObjectInputStream.java:2155
```

```
)
    at
java.io.ObjectInputStream.readOrdinaryObject(ObjectInputStream.java:
2013)
    at
java.io.ObjectInputStream.readObject0(ObjectInputStream.java:1535)
    at
java.io.ObjectInputStream.readObject(ObjectInputStream.java:422)
    at sun.rmi.transport.tcp.TCPEndpoint.read(TCPEndpoint.java:555)
    at sun.rmi.transport.LiveRef.read(LiveRef.java:292)
    at sun.rmi.server.UnicastRef2.readExternal(UnicastRef2.java:78)
    at
java.rmi.server.RemoteObject.readObject(RemoteObject.java:455)
    at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
    at
sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl
.java:62)
    at
sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAcce
ssorImpl.java:43)
    at java.lang.reflect.Method.invoke(Method.java:498)
    at
java.io.ObjectStreamClass.invokeReadObject(ObjectStreamClass.java:10
58)
    at
java.io.ObjectInputStream.readSerialData(ObjectInputStream.java:2122
)
    at
java.io.ObjectInputStream.readOrdinaryObject(ObjectInputStream.java:
2013)
    at
java.io.ObjectInputStream.readObject0(ObjectInputStream.java:1535)
    at
java.io.ObjectInputStream.defaultReadFields(ObjectInputStream.java:2
231)
    at
java.io.ObjectInputStream.readSerialData(ObjectInputStream.java:2155
)
    at
java.io.ObjectInputStream.readOrdinaryObject(ObjectInputStream.java:
2013)
    at
java.io.ObjectInputStream.readObject0(ObjectInputStream.java:1535)
    at
java.io.ObjectInputStream.readObject(ObjectInputStream.java:422)
    ... 15 more
```

Changes in 12cR1.4.3.10

The following changes were made in Oracle NoSQL Database 12cR1.4.3.10.

Topics

New Features in SQL for Oracle NoSQL

1. JSON data type and querying of JSON data.

   JSON can be used as the data type of a top-level table column, of a nested record field, or as the element type of an array or map. JSON is an abstract type: the actual value stored in a field of type JSON can be a valid json atomic value (integer, long, double, string, boolean, or the json null value), or an array, or a map. If the value is an array/map, the element type of the array/map will also be JSON. This implies that, unlike their fixed-schema counterparts, arrays and maps of JSON can contain heterogeneous elements.

   On input, JSON data is passed to Oracle NoSQL as a string or a text stream. In both cases, the text is parsed internally, and its constituent parts are converted to a tree of maps, array, and atomic value instances. Direct construction of JSON trees is also possible via APIs that construct and connect maps, array, and atomic values. For persistent storage, the tree is serialized into a binary format.

   JSON data is schema-less, in the sense that a field of type JSON can have very different kind of values in different table rows. For example, if "info" is a top-level table column of type JSON, in one row the value of "info" may be an integer, in another row an array containing a mix of doubles and strings, and in a third row a map containing a mix of other maps, arrays, and atomic values. Furthermore, the data stored in a JSON column or field, can be updated in any way that produces a still valid JSON instance. As a result, each JSON tree (either in main memory or as a serialized byte array on disk) is self-describing with regard to its contents.

   SQL for Oracle NoSQL can operate seamlessly on both data models: the strongly typed data supported in prior releases, as well as the schema-less JSON data. Tables can mix data from both models and all of the DML expressions can operate on both kinds of data. [#25455]

   Indexing JSON data is not supported yet. *[#25455], [#25436], [#25437], [#25727], [#25197], [#25731], [#25681], [#25683], [#25688], [#25559], [#25740], [#25629], [#25669], [#25671]*

2. New kinds of expressions in SQL for Oracle NoSQL.
   - NOT operator. Negates a boolean value. *[#25455]*
   - EXISTS operator. Checks whether an expression returns at least one value. *[#25455]*

- IS OF TYPE expression. Checks whether a value or a set of values has a specified data type. *[#25437]*
- CAST expression. Casts, if possible, a value or set of values to a specified data type. *[#25436]*
- Map constructor. Constructs a new map value. Both the field names and the field values may be computed by other expressions. Field values may be heterogeneous. *[#25455], [#25629]*
- CASE expression. Implements if-then-else semantics. *[#25197]*
- json null value and true/false literals. The (case-insensitive) keywords null, true, and false can be used as literals in a query. These are the only reserved keywords in SQL for NoSQL. DML operators and expressions were extended, if needed, to handle the new json null value. *[#25455], [#25731], [#25683]*

3. TIMESTAMP data type and values. A timestamp represents a point in time. It contains a date and a time. The date part has 3 components: year, month, and day of month. The time part has 4 components: hour, minute, second, and a decimal fraction of the second. The number of digits used to represent the fraction of a second is called the precision of the timestamp, and it can be a number between 0 (no fractional second) to 9 (nano second precision). Fields of timestamp types can be indexed and accessed in SQL queries. The SQL comparison operators have been extended to compare timestamps, and the cast operator can cast strings of appropriate formats to timestamp values. *[#24775], [#25707]*

Other Changes in SQL for Oracle NoSQL

Several of the pre-existing expressions in SQL for Oracle NoSQL have changed in this release, mostly to make such expressions more "JSON friendly". Some of these changes are backwards incompatible and applications may have to change their queries as a result.

1. Implicit array construction in SELECT clause If any expression in the SELECT list returns more than one values, an array is constructed to contain the values. *[#25455], [#25629]*
2. Several changes in path expressions, both syntactic and semantic.
   - Field steps accept atomic items as input. A field step returns an empty result if its input is an atomic item. Previously, an error would be raised in this case. *[#25455], [#25688]*
   - New syntax for filtering the entries of maps and records. *[#25455], [#25575], [#25740], [#25724]*

     Previously, square brackets ([<cond>?]) were used to filter the values of both maps and arrays. Furthermore, there was no way to filter map entries and return the qualifying map keys (instead of the values). Finally, there was no way to filter the entries of records. This release introduces the .keys(<cond>?) and .values(<cond>?) path steps. They are both supposed to work primarily with maps and records. .keys(<cond>?) selects the map/record entries that satisfy the condition (if any) and returns the keys (field names) of these entries. .values(<cond>?) selects the map/record entries that satisfy the condition (if any) and returns the field values of these entries. If the input is an array, .keys(<cond>?) and .values(<cond>?) are applied recursively on the array elements. If the input is an atomic item, the result is empty.

     This change also affects DDL statements that use paths, because the syntax for DDL paths is a subset of the DML syntax for paths. Most importantly,

CREATE INDEX statements must use the new syntax to index maps. Furthermore, applications that access multi-key map indexes directly, via the TableIterator APIs, must be updated to use the new path syntax to name the fields of the IndexKey instances they build to probe the indexes (alternatively, the IndexKey fields can be accessed by their position as well).

- Filtering and slicing arrays. The square brackets are still used to filter or slice arrays. They work the same way as before on arrays, except that the variable $elementPos has been renamed to $pos. However, in 4.3, the behavior is different if the input is not an array: in this case, an array is created on the fly, and the input item is inserted into that array; then, the filtering/slicing step operates on that array the same way as with any other array. *[#25455], [#25575], [#25740]*

3. Change in how the value comparison operators handle empty operands. Previously, if an operand returned the empty result, the result of the comparison would also be empty. In 4.3, if both operands return the empty sequence, the operands are considered equal (so true will be returned if the operator is =, <=, or >=). If only one of the operands returns empty, the result of the comparison is false unless the operator is !=. *[#25763]*
4. Change in how the value comparison operators handle incomparable values. Previously, an error would be returned in this case. In 4.3, false is returned as the result of the comparison. *[#25455], [#25681]*
5. Change in how the sequence-comparison operators handle the SQL NULL Previously, if any of the operand sequences contained NULLs, those NULLs would just be ignored. In 4.3, if a matching pair of items is not found, and either of the sequences contains a NULL, the result of the comparison is NULL (instead of false). *[#25455]*
6. Changes in array constructors. In 4.3, queries can construct arrays containing heterogeneous items. Furthermore, if an input expression returns an SQL NULL, the NULL is ignored (not inserted in the constructed array). Previously, both heterogeneous items and NULLs would cause an error to be raised. *[#25455], [#25730], [#25629]*

Other New Features

1. The information that is published to the <kvroot>/log/<storename>.stats and <kvroot>/log/<storename>.perf files is also available through the NoSQL JMX agent, via the standard javax.management.Notification mechanism. See the javadoc for oracle.kv.mgmt.jmx.StorageNodeMXBean for more information. *[#24979]*
2. The "makebootconfig" command now enables security by default. The `-store-security` flag is now optional, and has the value `enable` if not specified. You can still configure a non-secure store by specifying `none` as the value of this flag. In addition, the "kvlite" command now creates a secure store by default. You can create a non-secure store by specifying `-secure-config disable`. *[#25440]*

Bug and Performance Fixes

1. Strings used as keys in the MapValue type are now case-sensitive. Previously they were case-insensitive, which was a bug. Applications that assume case-insensitivity in MapValue keys may be affected. Field names in Record values remain case-insensitive, as documented. *[#25598]*

2. Tables created with map of json and array of json types will now work properly. Previously rows of these types would be created but could not be deserialized and retrieved. *[#25559]*
3. Removed the no-longer needed MapValue.putNull method and added methods on MapValue, ArrayValue, RecordValue and FieldDef, to allow construction of JSON null values for insertion into JSON maps and arrays. *[#25671]*
4. Fixed a problem where, if a table was created with fields other than primary key fields and it was evolved such that it contained only primary key fields, queries on the table could fail to return data. *[#25766]*
5. Fixed a problem that could result in a NullPointerException on a RepNode processing a query if a different RepNode were to become unavailable. *[#25792]*
6. Fixed a problem that could result in an ArrayIndexOutOfBoundsException when specifying a ReturnRow on a table where the primary key fields are not declared as the first fields of the table. *[#25819]*
7. Implemented compatibility with pre-4.3 DDL syntax for creating map indexes. Syntax allowed now includes:

   - path-to-map.keys() and path-to-map.values(), which are the current/new syntax.
   - KEYOF(path-to-map)
   - KEYS(path-to-map)
   - ELEMENTOF(path-to-map), path-to-map[]

   In addition, code has been added to handle pre-4.2 clients operating against 4.2 and later indexes that support null values. *[#25864]*
8. Added kvstore-ee.jar to the kvclient.jar manifest so that the Enterprise Edition works without the need to fix the classpath. This works in the Community Edition as well, where kvstore-ee.jar does not exist at all.
9. Fixed a bug that "create fulltext index if not exists" throws exception if the index exists, it should complete successfully but not require execution. *[#25664]*
10. Fixed a problem that could result in an ArrayIndexOutOfBoundsException when specifying a ReturnRow on a table where the primary key fields are not declared as the first fields of the table. *[#25819]*
11. Fixed a query bug where an "always false" filtering step, like `t.array[false].field`, would actually be evaluated as an "always true" step `(t.array[false].field)`. *[#25708]*
12. Fixed a query bug that would remove the entire FROM clause if the table name or table alias was not used anywhere in the rest of the query. This would cause the query compiler to throw an ArrayIndexOutOfBoundsException. *[#25768]*
13. Fixed a query bug that would cause a NPE when the table queried is a child table and is used without a table alias. *[#25645]*
14. Fixed a query bug that could cause an ArrayIndexOutOfBoundsException if the compiler could deduce that an expression would always return an empty result. For example, an ArrayIndexOutOfBoundsException would be thrown by this query: `select id[10:0] from foo f` *[#25673]*
15. Fixed a query bug: The fact the table names are case insensitive was not always taken into account in resolving column reference in a query. *[#25747]*

16. Fixed a bug that would cause an IllegalCommandException to be raise for a valid ALTER TABLE statement, when an index on an array or map existed on the same table. *[#25782]*
17. Fixed a query bug where a NULLS LAST directive in the order-by was being ignored, if the sort direction was DESC. Instead, an error should be thrown in this case. *[#25785]*
18. Added new methods to the oracle.kv.table.FieldValueFactory class for creating a TimestampValue from a Timestamp, String, or components: *[#25654]*

    ```
    public static TimestampValue createTimestamp(Timestamp v, int
    precision);
    public static TimestampValue createTimestamp(String s, int
    precision);
    public static TimestampValue createTimestamp(int year, int month,
    int day,
                                                  int hour, int minute,
                                                  int second, int nano,
                                                  int precision);
    ```

    Added new methods to the oracle.kv.table.TimestampValue interface to return the components of a Timestamp value:

    ```
    public int getYear();
    public int getMonth();
    public int getDay();
    public int getHour();
    public int getMinute();
    public int getSecond();
    public int getNano();
    ```

19. Fixed a bug that could result in failure in reading records from a table which contains a map/array field with min/max contraints on its element after upgrade from KV 3.0 to KV 4.3. *[#25799]*
20. Fixed a problem that could result in failure in table evolving when adding a new field that contains a nested fixed_binary, enum or record field. *[#25793]*
21. Changed query shell output default from COLUMN to JSON. *[#25700]*
22. Restricted numeric type mapping in JSON to integer, long, and double. This eliminates float as an option in JSON. *[#25699]*
23. Improved the error checking performed when processing DDL commands to provide clearer feedback when the requested command uses a feature that is not supported on all of the nodes in the store because of the nodes still need to be upgraded. *[#25712]*
24. Added checks to confirm that all shards have quorum and a majority of nodes in each zone being changed are online before deploying a topology that changes the type of one or more zones. This change makes it more likely that the topology change will detect and report, prior to making any changes, that, given the current lack of availability of certain replication nodes, the command will fail. *[#24503]*
25. Fixed upgrade problem where primary key fields marked as not nullable would cause serialization issues in queries causing them to fail. Primary keys can no

longer be explicitly marked a not nullable or with a default because they are implicitly not nullable and cannot have default values. *[#25533]*

26. Fixed a memory leak that could, in some cases, cause an OutOfMemoryError to be thrown when a replication node needs to do a Network Restore when performing replication. *[#25649]*
27. Improved the ability of the Admin CLI to validate new SN parameters when performing 'plan change-parameters' and 'change-policy' commands. With this change, commands that specify invalid SN parameters will fail without modifying the parameters, rather than having the incorrect parameters prevent SNs from being restarted. *[#25636]*
28. Fixed a problem in the Load utility that prevented it from restoring security information for a secure store. *[#24642]*
29. Fixed a problem that could prevent master rebalancing from working correctly because of excessive logging by reducing redundant debug logging entries. *[#25625]*
30. Fixed a problem in Hive and Big Data SQL query processing where a the SELECT clause from one query might affect the behavior of the next query if that query did not specify a SELECT clause. *[#25626]*

Storage Engine Changes (JE 7.3)

1. JE's low-level operation throughput statistics have been simplified and improved. Previously, these statistics represented API calls rather than CRUD operations and this caused confusion when a single API call performed multiple CRUD operations. Also, some CRUD operations (key search operations on any RN in a shard, all operations on a replica RN) were missing, and no operation statistics were included in the admin .stat files. The JE throughput stats were previously only visible via the env/je.stat.csv file.

   Now, the following statistics representing CRUD operations are output in the admin .stat files as well as env/je.stats.cvs, and are included for all RNs, including replicas. The new statistics should be considered internal operations or internal units of work. This approach is used to allow correlating internal operations to performance measurements. *[#23792]*

   ```
   priSearch
   priSearchFail
   secSearch
   secSearchFail
   priPosition
   secPosition
   priInsert
   priInsertFail
   secInsert
   priUpdate
   secUpdate
   priDelete
   priDeleteFail
   secDelete
   ```

2. Data corruption is now detected as soon as possible by a new internal JE background task. This detects data corruption caused by media/disk failure by reading the log sequentially and verifying checksums. This is the equivalent of

running the JE DbVerifyLog utility, but it is performed automatically and periodically on every RN.

By default, verification is on and occurs once a day at midnight, local time. Although this should not normally be necessary, verification can be disabled or the verification schedule can be changed using the JE EnvironmentConfig.ENV_RUN_VERIFIER, VERIFY_SCHEDULE and VERIFY_LOG parameters (je.env.runVerifier, je.env.verifySchedule and je.env.verifyLog).

When corruption is detected, the RN will be shut down and will not be restarted automatically by the SN. A SEVERE level exception will be logged in the RN's log and the exception message will contain the LOG_CHECKSUM token. Manual intervention by an administrator is necessary in this situation. We do not recommend restarting the RN without first replacing the media and/or forcing a network restore from another node in the shard.

The advantage of performing verification frequently is that a media/disk problem may be detected sooner than it would be otherwise. This means that the network restore can be done while the other RNs in the shard are up, minimizing exposure to additional failures. *[#25221]*

3. Repeat-fault reads have been eliminated for LNs (Btree leaf nodes, which represent record data on disk.) Previously, if an LN's on-disk size was greater than 4kB, two reads would be required to fetch the LN from disk. The first read would always include the log entry header, which contains the exact entry size, and the second read (repeat-read) was needed to read the entire entry. The second read includes the entire entry, although normally it will be cached by the file system.

   Now, only a single read is needed because the last logged size for LNs is now stored in the Btree, and this can be used to determine the exact size needed for the read buffer. The benefits of this change are:

   a. the amount of IO is reduced (although the repeat-read normally reads data that is cached by the file system)

   b. the statistics describing IO activity are simpler to analyze without the repeat-reads in the picture.

   Note that INs (Btree internal nodes) can still cause repeat-reads when they are fetched, because the last logged size for INs is not stored in the Btree. However, in most applications all INs are cached and therefore INs are rarely read from disk (except during a cache warm-up period). The nRepeatFaultReads statistic (EnvironmentStats.getNRepeatFaultReads) indicates the number of repeat-reads. *[#25387]*

4. A network restore (data copy from one RN to another RN in a shard) may be instigated for various reasons, for example, when a replica has been down for a long time period and is then brought up again. Previously, if a network restore was interrupted and the RN was then restarted, the RN could have attempted to operate with a partial data set, causing unpredictable results. Now, the use of a partially restored RN is not possible, and the network restore will automatically be restarted. If a network restore is interrupted, the RN's env directory will contain a marker file named 0x7fffffff.jdb, which is recognized and managed by JE. *[#25369]*

5. Fixed a bug that prevented the request timeout for a write operation from being honored. Under certain circumstances, the request took longer than the specified timeout. *[#25692]*

Deprecated Features

1. The NONE_REQUIRED_NO_MASTER (oracle.kv.Consistency.NONE_REQUIRED_NO_MASTER) consistency policy that requires that a read operation be serviced on a replica, never the Master, is deprecated in this release. Applications should use the NONE_REQUIRED consistency policy or KVStoreConfig.setReadZones instead.

Changes in 12cR1.4.2.14

The following changes were made in Oracle NoSQL Database 12cR1.4.2.14.

Topics

New Features

1. This release introduces the Basic Edition. Basic Edition (BE) only contains the server and is distributed under the Oracle NoSQL Database Enterprise Edition license. The Basic Edition is functionally identical to Oracle NoSQL Database Community Edition. *[#25663]*

Bug and Performance Fixes

1. Fix a bug that was introduced in release 4.2.10 that results in failures when executing some Admin commands. The bug arises after an upgrade, when the store has explicit storage directories. These are the directories used by Replication Nodes and are specified via the `plan change-storagedir` CLI command or the `makebootconfig` utility. After the upgrade the `verify configuration` and some `plan` CLI commands will fail, with an error reporting that a parameter cannot be represented as long. *[#25689]*
2. Fix a bug in the EXPORT utility where it no longer hangs when the number of tables exceeds the fixed number of threads allocated for the operation. *[#25717]*
3. Oracle NoSQL Database server will only verify the major and minor versions of Oracle Java SE and not the patch version at `makebootconfig` time. *[#25647]*
4. Primary keys are implicitly not nullable but also have no default values. Marking primary key fields as not nullable or with default values is no longer allowed and the table schema reflects that. This also fixed an upgrade issue related to allowing this in earlier releases. *[#25533]*

Changes in 12cR1.4.2.10

The following changes were made in Oracle NoSQL Database 12cR1.4.2.10.

Topics

New Features

1. The topology builder has been enhanced to take storage directory sizes into account when laying out shards and partitions. What this means is a RepNode with larger storage directory will be assigned a larger portion of the store's data than a RepNode with smaller directory. Storage directory size layout will happen automatically during elasticity operations when directory sizes have been configured. To enable this feature on an existing store the size of storage directory for each RepNode needs to be set, usually by the administrator. Then a rebalance, redistribute, or contract topology operation will adjust the shards and/or partitions to account for differences in directory sizes.

   To set or change the directory size on existing stores there is a new `-storagedirsize` flag on the `plan change-storagedir` Admin CLI command. To set the directory size on a new configuration the flag `-storagedirsize` has been added to `makebootconfig`. Both flags can accept size values with units, for example: "1_TB" can be used to specify a 1 terabyte directory. *[#24981], [#25166]*
2. DDL operations are now logged with the standard audit logging prefix, "*timestamp* KVAuditInfo", to support searching and filtering. *[#25460]*
3. Added Arbiter functionality. Using this feature, KVStore DML operations using Durability.ReplicaAckPolicy.SIMPLE_MAJORITY durability succeeds even with one RepNode node unavailable in a shard. The data written with "relaxed" durability will be migrated to the other RepNode when it becomes available. Also, only the node with the "relaxed" durability data can become master until the data is migrated to the other RepNode. This fact relaxes the write availability to a single node failure as opposed to write availability with any node failure (on a shard basis). *[#20590]*
4. Added the new 'topology contract' command to the Administrative CLI. This command provides support for removing a specified set of storage nodes from the topology and shrinking the size of the store by removing shards. Also added the new 'pool leave' command to simplify the process of specifying which storage nodes should be removed. *[#24425]*
5. Secondary indexes now contain entries for rows that have null values for index fields. This serves two purposes:
   a. It means that an index will have an entry for every row in the table, regardless of values, which was not true previously

b. It makes composite indexes (those with multiple fields) more useful because an application can use a partial key and know that rows will not be skipped because of a null in another field

This change required a modification of the format of the index databases themselves. All new indexes will have this new format and will support null values. Existing indexes cannot support null values and will continue to operate as they have in the past, without nulls. For this reason, it is recommended that indexes be dropped and re-added if null values are desired. *[#24785]*

6. Added new policy parameters that permit administrators to enforce password complexity requirements when users create a new passwords or change existing passwords. *[#24985]*
7. Added new position-based put and get methods on RecordValue that work more efficiently than the name-based methods if the application knows the position of a field within a RecordValue. *[#25214]*
8. Includes a preview release of Java API support for a field type of JSON. Use of this type is for preview only. Any stores that use this type will *not* be supported, and must be removed and not re-used when full support is provided. This preview includes: *[#23589]*

   - The ability to declare a field as type JSON. Doing so means that any type that can be interpreted as valid JSON can be used in this field, including the numeric types, boolean, string, map (JSON object), array (JSON array).
   - Methods on FieldDef and FieldValue and related types to put and get JSON values as well as to navigate into JSON fields in a Row.

   The preview release has the following known limitations and issues, which can result in failures or unpredictable results:

   - Indexes into JSON are not yet supported. This is a feature that will be supported in a future release.
   - Queries involving JSON are not yet supported, and will fail to compile and/or execute. This is a feature that will be supported in a future release
   - Not all numeric values in JSON are supported. Any number that cannot be represented as a long or double will fail to be handled. This feature will be supported in a future release.
   - Map keys are case-insensitive, which extends to JSON objects. This is a bug and will be fixed in a future release.
   - Support for declarations of MAP(JSON) and ARRAY(JSON) in fields is not complete. These declarations will work in DDL statements, but rows in such tables will not be usable and must be avoided. This is a bug and will be fixed in a future release. release.

Bug and Performance Fixes

1. Enhanced the Load utility to use a disk-ordered cursor to read entries from multiple databases of snapshot. This change improves the input throughput, thus improving the overall performance of loading. *[#25294]*
2. Fixed a problem where specifying a large value for the rnHeapMaxMB storage node parameter could result in the replication nodes hosted by that storage node being given heap sizes that do not support compressed object references, even though a smaller heap size that supports compressed object references would be more efficient. *[#25472]*

3. Fixed a bug that the TextIndexFeeder failed to act accordingly when the topology changed and partition migrated between replication groups. The fix ensures that the TextIndexFeeder will stream all writes to Elasticsearch when the store undergoes elasticity operations. *[#25334]*
4. Fixed a problem where an unexpected InsufficientAcksException caused by a lack of shard quorum during an access control change could cause a replication node to be restarted. *[#25439]*
5. Fixed a problem where a lack of shard quorum during an access control change could result in a temporary deadlock and an unexpected InsufficientAcksException. *[#25442]*
6. Fixed a problem where an upgrade of a 3.x version store would fail if the store contained a table that previously had been schema evolved. *[#25532]*
7. Disable full text search (FTS) in secure store. If a KV store is configured as a secure store, FTS will be disabled and user will not be allowed to register an external Elasticsearch cluster. In addition, for a KV store in which FTS is already enabled, user will not be able to re-configure the store from a non-secure to a secure store. Instead, she need to drop all text indices, deregister the external Elasticsearch cluster, and re-configure the non-secure store to a secure store. *[#25245], [#25246]*
8. RepNodes are now configured to use the Java option -XX:+AlwaysPreTouch by default on Linux platforms when using the Oracle Java virtual machine. This change slows Java startup slightly (roughly 10 sec for a 32GB heap using 4K pages and less than 1 sec if using 2MB Large pages), but reduces subsequent latency pauses as larger amounts of heap storage are used by the RepNode. *[#25161]*
9. RepNodes now use the Garbage First Garbage Collector (G1 GC) by default. The G1 GC typically provides shorter GC pause times than the Concurrent Mark Sweep (CMS) collector, which was the previous default. As a result, the G1 GC should reduce both average and peak latency for store operations, and improve throughput.

   Applications can revert to the previous GC settings using the CMS GC by including `-XX:+UseConcMarkSweepGC` in the value of the RepNode `javaMiscParams` parameter. *[#24695]*
10. The representation of IndexKey has been changed in an incompatible manner. The reason is a combination of ease of use, function, and performance. Prior to this release an IndexKey shared schema (the RecordDef) with the corresponding Row objects for the same table. The old representation was confusing and cumbersome in the face of indexes on fields in deeply nested records, maps, and arrays. The new representation is a *flattened* version of the fields indexed, with only a single level of structure, where the field name is a path to the indexed field.

    Consider this table and index:

    ```
    CREATE TABLE user (id INTEGER, PRIMARY KEY(id), address RECORD(city
    String, state String))
    CREATE INDEX City on user(address.city)
    ```

Prior to this release the Java code required to create an IndexKey used to iterate the City index would look like this:

```
/* assume userTable is a handle on the table */
Index index = userTable.getIndex("City");
IndexKey indexKey = index.createIndexKey();
indexKey.createRecord("address").put("city", "Chicago");
```

Note the need to create the *structure* of the record to use it. The new flattened representation is:

```
/* assume userTable is a handle on the table */
Index index = userTable.getIndex("City");
IndexKey indexKey = index.createIndexKey();
indexKey.put("address.city", "Chicago");
```

The syntax for indexes involving maps and arrays is as follows:

- Map keys: *keys(path-to-map-field).* For example *keys(this_is_a_map)*
- Map values: *path-to-map-field[].* For example, *this_is_a_map[].* If the index is on a field within a map of records it would look like *map_of_records[].path-to-field*
- Array values: these are similar to map values and require use of *"[]": path-to-array[].*

The same syntax rules are used for these paths as are used in the statements that create the indexes themselves. Similar syntax works for indexes on other complex types such as maps and arrays.

This is an incompatible change and will require any applications that use indexes on complex types to be modified to use the new format and syntax. If unchanged, errors will be seen as IllegalArgumentException thrown from IndexKey.put*() calls as well as FieldRange construction for such fields. Applications that do not use indexes on records, maps, or arrays will continue to work without modification. See the documentation for details on how to represent paths to complex index fields. *[#25090]*

11. Changed to allow an existing client to access store without reopening a KVStore handle during updating SSL certificate of a secure store. *[#25062]*
12. The exception, KVSecurityException, extended FaultException, which was not correct, according to the contract for FaultException. KVSecurityException now extends RuntimeException.

    This is a minor API change but it affects the possible exceptions thrown from most of the methods on the KVStore and TableAPI interfaces. They can now throw KVSecurityException, which no longer falls under the umbrella of FaultException. Applications that need to catch one or more of the KVSecurityException instances — UnauthorizedException, AuthenticationRequiredException, and AuthenticationFailureException — need to do so explicitly. *[#24967]*
13. To make text indexing more efficient, index population now uses Elasticsearch's bulk operations interface. *[#25040]*
14. Duplicate results from map and array indexes have been eliminated. Previously, when performing a query involing map value or array indexes, it was possible to see duplicate results for rows that have multiple independent entries in the index.

This would occur when using the index iteration methods as well. These duplicates are now removed so that the user will see only one instance of each matching record. *[#25023]*

15. To avoid a compatibility issue, NoSQL Database by default now connects to an Elasticsearch cluster via Elasticsearch's Transport Client rather than its Node Client. [#25146]
16. Records that contain only empty strings in text-indexed fields are now correctly added to the text index. Previously, such records were omitted. *[#25058]*
17. Two race conditions that could occur during text index creation have been fixed. *[#25093], [#25182]*
18. The size of the service port range required for storage nodes has been reduced. Previously, when -servicerange was used when calling makebootconfig to configure a storage node that hosts an admin, the range required 8 ports for a non-secure deployment. That number has now been reduced to 3. For details, see the documentation for the servicePortRange parameter in the section on Storage Node Parameters in the Admin Guide. *[#24708]*
19. Allow the anonymous user to use a DDL statement in the Admin CLI to create the first user for a secure store. *[#25051]*
20. Fixed a problem where a client with an incorrect set of helper hosts could cause replication nodes to fail and not restart. The problem could occur if the client was configured with helper hosts from unrelated stores and attempted to forward topology updates obtained from one store to another store. Such an occurrence is now logged but no longer causes the replication node to fail. *[#24693]*
21. Changed the maximum value of the memoryMB parameter from 500 GB to 128 TB, to accommodate machines with large amounts of memory. *[#25017]*
22. Fixed a problem in the statistics gathering code that could cause a replication node to fail because of an unexpected InterruptedException: *[#25046]*

```
2016-04-12 13:34:42.678 UTC INFO [rg1-rn3] Exception accessing the
migration db {0}
com.sleepycat.je.ThreadInterruptedException: (JE 7.0.5) Environment
must be closed, caused by:
com.sleepycat.je.ThreadInterruptedException: Environment invalid
because of previous exception: (JE 7.0.5) rg1-rn3(3):/scratch/yfei/
nftest/dm_mode/kv_isolate_rn/scratch/kvroot/mystore/sn3/rg1-rn3/env
java.lang.InterruptedException THREAD_INTERRUPTED:
InterruptedException may cause incorrect internal state, unable to
continue. Environment is invalid and must be closed.
       at
com.sleepycat.je.ThreadInterruptedException.wrapSelf(ThreadInterrupt
edException.java:135)
       at
com.sleepycat.je.dbi.EnvironmentImpl.checkIfInvalid(EnvironmentImpl.
java:1720)
       at
com.sleepycat.je.Transaction.checkEnv(Transaction.java:886)
       at com.sleepycat.je.Transaction.commit(Transaction.java:350)
       at
oracle.kv.impl.rep.migration.PartitionMigrations.fetch(PartitionMigr
ations.java:295)
       at
```

```
oracle.kv.impl.rep.migration.MigrationManager$3.call(MigrationManage
r.java:1286)
       at
oracle.kv.impl.rep.migration.MigrationManager$3.call(MigrationManage
r.java:1282)
       at
oracle.kv.impl.rep.migration.MigrationManager.tryDBOperation(Migrati
onManager.java:1464)
       at
oracle.kv.impl.rep.migration.MigrationManager.getMigrations(Migratio
nManager.java:1282)
       at
oracle.kv.impl.rep.migration.MigrationService.pendingSources(Migrati
onService.java:180)
       at
oracle.kv.impl.rep.migration.MigrationManager.awaitIdle(MigrationMan
ager.java:445)
       at
oracle.kv.impl.rep.table.MaintenanceThread.run(MaintenanceThread.jav
a:149)
```

23. Fixed an unhandled security exception in the master rebalancing code that could cause a replication node to fail unexpectedly when a storage node is being restarted. *[#25134]*

24. Fixed a problem where a replication node in a secure store could exit with a NullPointerException if it received a request before it is fully initialized. *[#25092]*

```
2016-05-03 18:08:42.191 UTC SEVERE [rg5-rn3] Process exiting
java.lang.NullPointerException
at
oracle.kv.impl.rep.login.KVSessionManager.initializeKVStore(KVSessio
nManager.java:495)
at
oracle.kv.impl.rep.login.KVSessionManager.isReady(KVSessionManager.j
ava:259)
at
oracle.kv.impl.rep.login.KVSessionManager.resolve(KVSessionManager.j
ava:425)
at
oracle.kv.impl.security.login.TokenResolverImpl.resolvePersistentTok
en(TokenResolverImpl.java:216)
at
oracle.kv.impl.security.login.TokenResolverImpl.resolve(TokenResolve
rImpl.java:168)
at
oracle.kv.impl.security.login.TokenVerifier.verifyToken(TokenVerifie
r.java:97)
at
oracle.kv.impl.security.AccessCheckerImpl.identifyRequestor(AccessCh
eckerImpl.java:141)
at
oracle.kv.impl.security.ExecutionContext.create(ExecutionContext.jav
a:181)
at
```

```
oracle.kv.impl.security.SecureProxy$CheckingHandler$1.execute(Secure
Proxy.java:609)
at
oracle.kv.impl.security.SecureProxy$CheckingHandler$1.execute(Secure
Proxy.java:600)
at
oracle.kv.impl.fault.ProcessFaultHandler.execute(ProcessFaultHandler
.java:148)
at
oracle.kv.impl.rep.admin.RepNodeAdminFaultHandler.execute(RepNodeAdm
inFaultHandler.java:124)
at
oracle.kv.impl.security.SecureProxy$CheckingHandler.invoke(SecurePro
xy.java:598)
at oracle.kv.impl.security.SecureProxy.invoke(SecureProxy.java:144)
at com.sun.proxy.$Proxy9.getTopoSeqNum(Unknown Source)
at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
at
sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl
.java:62)
at
sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAcce
ssorImpl.java:43)
at java.lang.reflect.Method.invoke(Method.java:498)
at
sun.rmi.server.UnicastServerRef.dispatch(UnicastServerRef.java:323)
at sun.rmi.transport.Transport$1.run(Transport.java:200)
at sun.rmi.transport.Transport$1.run(Transport.java:197)
at java.security.AccessController.doPrivileged(Native Method)
at sun.rmi.transport.Transport.serviceCall(Transport.java:196)
at
sun.rmi.transport.tcp.TCPTransport.handleMessages(TCPTransport.java:
568)
at
sun.rmi.transport.tcp.TCPTransport$ConnectionHandler.run0(TCPTranspo
rt.java:826)
at
sun.rmi.transport.tcp.TCPTransport$ConnectionHandler.lambda$run$0(TC
PTransport.java:683)
at java.security.AccessController.doPrivileged(Native Method)
at
sun.rmi.transport.tcp.TCPTransport$ConnectionHandler.run(TCPTranspor
t.java:682)
at
java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor
.java:1142)
at
java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecuto
r.java:617)
at java.lang.Thread.run(Thread.java:745)
```

25. Added TableOperationResult.getPreviousExpirationTime() to return the expiration time of a previous row, if defined. Previously the expiration time was available from the previous Row, which was a partial solution. *[#25229]*

26. Fixed a problem that client may push update of topology without signature against a secure store. Before this fix, server audit logging may have warning entries like: *[#25273]*

    ```
    2016-06-13 06:53:32.476 UTC SEC_WARNING [rg2-rn1] Empty signature.
    Verification failed for topology seq# 38
    ```

27. Fixed a bug where an application that used a KVStore handle from an old, defunct store, against a new, recreated store on the same hosts with the same ports could cause replication node restarts in the new store. Now the store is unaffected and the application will receive an oracle.kv.StaleStoreHandleException, letting it know that the handle should be closed and reopened. *[#24693]*

Utility Changes

1. If users need to configure Kerberos manually, for example when using Active Directory, they can now specify "none" for the "-kadmin-path" argument in the makebootconfig and securityconfig commands. In this case, the keytab is not automatically generated and must be generated and copied manually. *[#25445]*
2. The `verify configuration` Administrative CLI command now shows a warning if a failed deployment resulted in a store topology with shards that have no partitions. *[#22098]*
3. Added security configuration utilities: *[#24948]*
   - security config update

     Update security parameters in given security configuration.
   - security config verify

     Verify the consistency and correctness of given security configuration.
   - security config show

     Print out all information of given security configuration.
4. Enhanced "oracle.kv.util.Load" utility to support loading data from multiple shard backup directories concurrently, and use bulk put to improve performance of loading data. *[#25085]*
5. Running the admin web service is no longer supported for secure stores, which results in the following changes to utilities: *[#25249]*
   - Using the `makebootconfig` command to configure a storage node for a secure store must specify `-runadmin` if the storage node should run an admin, and either specify the admin port as 0 or leave the admin port unspecified
   - An admin deployed on a secure store with `plan deploy-admin` must specify 0 for the value of the `-port` flag
   - The `securityconfig add-security` command fails when performed against a non-secure store that has an admin with an admin web service
   - The `plan change-parameters` command fails when attempting to specify a non-zero admin port on a secure store
   - The `plan migrate-sn` command fails when attempting to migrate a storage node with a non-zero admin port on a secure store

Storage nodes in a secure store that run an admin with the admin web service enabled will not start successfully after upgrading to this release. Make sure to disable the admin web service before performing an upgrade.

If you attempt to upgrade a storage node for a secure store that has the admin web service enabled, the node will fail to start with a message like:

```
Failed to start SNA: Cannot start the storage node agent for a
secure
store when the storage node has an admin with the admin web service
enabled.  Please start this storage node and reconfigure the storage
node to disable the admin web service, or disable store security
with
12.1.4.0.9 before starting this storage node.
```

If you see this message when starting a storage node, you can work around the problem by reverting to the previous release software and restarting the storage node. Then, use the `plan change-parameters` command to disable the admin web service. For example, to disable the web service on admin1, you could use the command:

```
plan change-parameters -service admin1 -wait -params adminHttpPort=0
```

SQL Query Language and Shell Changes

1. The Oracle NoSQL Database SQL query language has been enhanced with the following features:
   - Support for OFFSET and LIMIT *[#25078]*
   - ALTER TABLE, used for schema evolution, has been modified to allow modification of RECORD types nested within other records, maps, and arrays. *[#24049]*
2. Paths used in DDL statements to reference indexes involving arrays, map keys, and map values have been modified to be consistent with those used in queries. The statements affected include `CREATE INDEX` and `DESCRIBE TABLE`. *[#25167]*

   There are 3 types of paths:

   a. Paths to array values.

   ```
   Previous syntax: path-to-array or path-to_array[]
   New syntax: path-to-array[] ("[]" is required)
   For example, to create an index on the values in an array:
       CREATE TABLE MyTable(id INTEGER, myArray ARRAY(INTEGER),
   PRIMARY KEY(id))
       CREATE INDEX ArrayIndex on MyTable(myArray[])
   ```

   b. Paths to map values:

   ```
   Previous syntax: path-to-map or path-to-map[]
   New syntax: path-to-map[] ("[]" is required)
   For example, to describe the field used in a map:
       CREATE TABLE MyTable(id INTEGER, myMap MAP(STRING), PRIMARY
   ```

```
KEY(id))
    DESCRIBE AS JSON MyTable(myMap[])
```

c. Paths to map keys:

```
Previous syntax: KEYOF(path-to-map)
New syntax: KEYS(path-to-map)
For example, to create an index on the keys of a map:
    CREATE TABLE MyTable(id INTEGER, myMap MAP(STRING), PRIMARY
KEY(id))
    CREATE INDEX MapIndex on MyTable(myMap[])
```

3. The **CREATE FULLTEXT INDEX** and **DROP INDEX** DDL statements have acquired new syntax to allow overriding of the new constraints. *[#24809]*
4. New constraints are now enforced when text indexes are created and deleted. These operations are not allowed unless the health status of the Elasticsearch cluster is *GREEN*. *[#25093]*
5. Fixed a bug where executing a query statement that contains != condition in the SQL shell could result in a java.io.IOException: Invoke method readLine of Jline.ConsoleReader failed: !=... : event not found. *[#25065]*
6. Added a command "show query <statement>" to display the query plan for a query to the SQL shell. This is not part of the query language itself; it is a feature of the SQL shell. *[#25170]*

Storage Engine Changes (JE 7.2)

1. Fixed a problem where checkpointing sometimes did not occur after log cleaning when application write operations stopped, which prevented reclamation of disk space. This was a common problem with tests that expect disk space to be reclaimed after write operations have stopped. In production systems it could also be a problem during repair of an out-of-disk situation. Note that an earlier fix *[#23180]* in JE 7.1 caused cleaning to occur in this situation, but a checkpoint is also needed to reclaim disk space, so the earlier fix was incomplete. *[#25364]*
2. Unexpected JE data file (.jdb file) deletions are now automatically detected by a background task. Normally all JE data file deletions should be performed internally as a result of JE log cleaning. If an external file deletion is detected, JE assumes this was accidental. This will now cause the RN to fail very quickly, so that the problem is made visible as soon as possible. Previously, the problem could be undetected for a period of time if the deleted file was not frequently accessed. *[#25201]*
3. Further reduced the possibility that multiple node failures could cause the loss of data (JE's RollbackProhibitedException). When NO_SYNC durability is used, JE flushes the log to disk periodically to avoid this. Previously, an fsync (flush to the storage device) was performed every 5 minutes. Now, a flush to the file system is performed every 5 seconds and an fsync is performed every 20 seconds. *[#25417]*
4. JE's DbCacheSize utility has been improved for applications like NoSQL DB that use CacheMode.EVICT_LN and an off-heap cache. The -offheap argument should be specified when running DbCacheSize during NoSQL DB capacity planning. The documentation for running DbCacheSize has been updated in the Determine JE Cache Size section of the NoSQL DB Administrator's Guide, C. Initial Capacity Planning. This documentation was previously incorrect because it did not take into account use of the off-heap cache. *[#25380]*

Deprecated Features

1. The SNMP agent (oracle.kv.impl.mgmt.snmp.SnmpAgent), which makes store metrics available for access via the SNMP protocol, is deprecated in this release. It will be removed from the product in a future release, circa July 2017.

Changes in 12cR1.4.0.9

The following changes were made in Oracle NoSQL Database 12cR1.4.0.9.

Topics

New Features

1. Added the ability to define a Time To Live (TTL) on an individual table row. The TTL can be defined in terms of hours or days and can be declared as a default on a table or applied per-row on put operations. Applying a TTL to a row results in an absolute expiration time on that row after which it will not be available. The expiration time is available using new API on Row. *[#24743]*
2. Includes a preview release of an SQL-like declarative query language, called ONQL, that can be used to access data in tables in a read-only fashion. As a preview release its use and feedback is encouraged. We expect that additions and changes done for a later formal release will be backward-compatible with this preview, however we cannot rule out minor incompatibilities.

   As of this preview release the language supports SELECT-FROM-WHERE-ORDERBY queries that can perform projection, predicate-based filtering, simple arithmetic operations, index-based sorting, and path expressions for navigating and projecting out data from complex structures (records, arrays, and maps). Bind variables are also supported for reuse of prepared queries.

   The ONQL processor performs all operations on the server side and automatically takes advantage of primary and secondary indexes to increase query performance. The language also supports hints to allow users to explicitly choose an index for a query, in cases where the query processor is not making the best choice among the multiple indexes applicable to a query.

   In addition a new ONQL shell has been added to facilitate use of the query language in an interactive, command line interface. *[#24186]*
3. Added Export/Import utility. Export utility allows a user to export the contents from Oracle NoSql Store to an external export store (Local File System / Oracle Storage Cloud Service). Import utility allows a user to import the contents from an external export store (Local File System / Oracle Storage Cloud Service) to Oracle NoSQL store. *[#24734]*
4. Added a preview release of integration with Elasticsearch to provide full text search indexing. Text indexes can be added to tables in Oracle NoSQL Database, which will cause a corresponding index in an attached Elasticsearch cluster to be populated and maintained. *[#23733]*

   To use this feature, users must first make available on the network an Elasticsearch 2.0 cluster, and register the cluster with the NoSQL Store. This enables use of the CREATE FULLTEXT INDEX DDL command.

5. Added mechanism to support creating statistics tables automatically at the end of the existing plan deploy-topology. *[#24768]*
6. Added a *predicate pushdown* mechanism to the Oracle NoSQL Database Table API Hive Integration classes. Specifically, a mechanism was added which supports the decomposition of a Hive or Big Data SQL query's WHERE clause (the predicates) into information that can be passed to the KVStore database so that some/all search processing can be performed in the database itself rather than in the processing performed on the client side of the query. *[#24525]*
7. Added a non-interactive login mode for command line utilities like Admin command line interface and Ping utility. Setting the login property introduced this release, "oracle.kv.password.noPrompt" as true, the non-interactive login mode will be enabled. If users are not using password store and configuring login properties incorrectly, the utilities would directly exit without automatically prompting password or user name like previous releases. *[#24931]*
8. Added a new example that demonstrates how to perform Kerberos authentication on the client side. *[#24328]*
9. Ranges declared on fields in DDL "CREATE TABLE" expressions are no longer supported. In previous releases it was possible to declare range constraints on fields in tables. This was done using the data definition language (DDL) keyword "CHECK" and providing a range. The primary intent of this feature was to allow the system to optimize primary key size for such fields because keep primary key size small is an important optimization. This feature was allowed for numeric and String types.

   As of this release the CHECK syntax and generalized ranges are no longer supported. Existing tables created with such fields will continue to operate correctly and enforce the range constraints. To replace this function, an optional storage size specification is allowed on fields declared in the primary key. This makes the intent of the feature very clear — reducing primary key size. This is only allowed on fields of type INTEGER at this time. *[#24769*]
10. Added new methods to support bulk put operations in the table API. See the documentation for the TableAPI.put(List<EntryStream<Row>>, BulkWriteOptions) and KVstore.put(List<EntryStream<KeyValue>>, BulkWriteOptions) methods for more details. *[#24563]*

Bug and Performance Fixes

1. Changes have been made to the Admin and Rep Node service's persistent store to support more flexible upgrades in future releases. This change will have a one-time impact when upgrading from a previous release to this release. During the upgrade the Admin may temporarily enter a "read-only" mode. While in the read-only mode it will not be possible to change the persistent state of the Admin. This includes creating new topologies, changing parameters, creating new plans, or running existing plans. A read-only Admin will attempt to exit this mode by transferring control to another Admin node. *[#24634], [#24725]*
2. System tables that are internally managed and maintained by the system are now read-only when security is enabled on the store. In addition, whether or not security is enabled, the schema for system tables are now immutable, and users are not permitted to alter, or drop or add indexes for system tables. The old tables (TableStatsPartition, TableStatsIndex, PartitionStatsLease and IndexStatsLease) used by statistics gathering are deprecated in this release. If enabled statistics gathering in an earlier release, you are still able to read these tables to check old

statistics data, but they will not be updated in this release. You can drop old tables if you do not have any plan to maintain old statistics. *[#24467]*

3. Fixed a problem that prevented the ping command from authenticating properly when used with a secure store. *[#24808]*

Utility Changes

1. Due to changes in the Admin service's persist store (described in the "Bug and Performance Fixes" section above) the new versions of the dump and load utilities can only be used after the store has been fully upgraded. Also, previous versions of these utilities will no longer work once an upgrade to this release has begun. During an upgrade the new utilities will not operate until all of the Admin nodes have been upgraded and the Admin is no longer in the read-only mode described above. *[#24634], [#24725]*
2. The "table size" sub command was mistakenly marked to be deprecated together with its parent command "table" command in release 12cR1.3.5.2, move "table size" to top level command and rename it to "table-size".
3. Added a new status utility command that shows the status of a storage node agent. *[#23561]*
4. Fixed a problem where executing a Hive query against a table containing an array with no elements would cause an exception like the following: *[#24900]*

```
2016-02-09 13:31:24,213 ERROR [main]: CliDriver
(SessionState.java:printError(921)) - Failed with exception
java.io.IOException:java.lang.IndexOutOfBoundsException: Index: 0,
Size: 0
java.io.IOException: java.lang.IndexOutOfBoundsException: Index: 0,
Size: 0
at
org.apache.hadoop.hive.ql.exec.FetchOperator.getNextRow(FetchOperato
r.java:507)
at
org.apache.hadoop.hive.ql.exec.FetchOperator.pushRow(FetchOperator.j
ava:414)
at
org.apache.hadoop.hive.ql.exec.FetchTask.fetch(FetchTask.java:138)
at org.apache.hadoop.hive.ql.Driver.getResults(Driver.java:1655)
at
org.apache.hadoop.hive.cli.CliDriver.processLocalCmd(CliDriver.java:
227)
at
org.apache.hadoop.hive.cli.CliDriver.processCmd(CliDriver.java:159)
at
org.apache.hadoop.hive.cli.CliDriver.processLine(CliDriver.java:370)
at
org.apache.hadoop.hive.cli.CliDriver.executeDriver(CliDriver.java:75
6)
at org.apache.hadoop.hive.cli.CliDriver.run(CliDriver.java:675)
at org.apache.hadoop.hive.cli.CliDriver.main(CliDriver.java:615)
at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
at
sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl
.java:62)
```

```
at
sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAcce
ssorImpl.java:43)
at java.lang.reflect.Method.invoke(Method.java:497)
at org.apache.hadoop.util.RunJar.run(RunJar.java:221)
at org.apache.hadoop.util.RunJar.main(RunJar.java:136)
Caused by: java.lang.IndexOutOfBoundsException: Index: 0, Size: 0
at java.util.ArrayList.rangeCheck(ArrayList.java:653)
at java.util.ArrayList.get(ArrayList.java:429)
at
oracle.kv.impl.api.table.ArrayValueImpl.get(ArrayValueImpl.java:97)
at
oracle.kv.hadoop.hive.table.TableSerDe.objectInspector(TableSerDe.ja
va:458)
at
oracle.kv.hadoop.hive.table.TableSerDe.deserialize(TableSerDe.java:2
86)
at
org.apache.hadoop.hive.ql.exec.FetchOperator.getNextRow(FetchOperato
r.java:488)
... 15 more
```

Deprecated Features

1. The interfaces and classes related to support of Avro have been deprecated in favor of the use of tables and the table API. All Avro features are available more simply using the table API. The interfaces and code supporting Avro will continue remain available for some time, but users are encouraged to migrate to the table API.

   The interfaces and classes affected are mostly in the oracle.kv.avro package. Interfaces that use those classes are also affected.

Changes in 12cR1.3.5.2

The following changes were made in Oracle NoSQL Database 12cR1.3.5.2.

Topics

New Features

1. Added support for gathering key distribution statistics. To enable statistics gathering, the statistics tables need to be created first by executing the runadmin script file `CreateStatisticTables.kvs`, which is located in the `lib` directory of the release package. *[#23422]*
2. Added support for Kerberos user authentication. Users that wish to use this new capability should be aware of the following changes:
   - The makebootconfig utility has a new optional `-external-auth` command-line argument which is used to enable external authentication mechanisms. Currently, Kerberos is the only supported mechanism. This new flag is only permitted when the value of the `-store-security` flag is specified as `configure` or `enable`.
   - The securityconfig tool has been enhanced to provide the ability to:
     - Add Kerberos configuration information to an existing security configuration
     - Renew a store's Kerberos service principal keytab for security maintenance tasks
   - A new class, KerberosCredentials, which implements LoginCredentials, has been introduced, and its instances can be used in the KVStoreFactory.getStore method to perform a Kerberos login:
     - `KVStoreFactory.getStore(KVStoreConfig, LoginCredentials, ReauthenticateHandler)`
   - New Kerberos-related security properties have been added that must be specified when using a KVStore command-line utility program against a secure store that has enabled Kerberos authentication.

   There are two approaches that applications can use to authenticate using Kerberos. Client applications that use the Java Authentication and Authorization Service (JAAS) programming framework can specify credentials by using `Subject.doAs`. Applications that do not use the JAAS framework can use the new security properties to specify parameters needed to acquire user credentials from the Kerberos Key Distribution Center (KDC).

   Kerberos authentication is described in greater depth in the Oracle NoSQL Database Security Guide, as well as in the Administrators Guide and the Javadoc. *[#24328]*
3. The off-heap cache is now enabled by default with a systemPercent setting of 10%. Please review Managing the Off-heap Cache for details.

Bug and Performance Fixes

1. In earlier versions, a LOB read operation with `Consistency.NONE_REQUIRED` could fail in the absence of a master for the shard hosting the LOB metadata. The read operation will now use a suitable replica when the read consistency allows for it. *[#24460]*
2. Fixed a bug where a client calling TableAPI.getTable() while an index is being created and populated (is not yet ready) could result in a ConcurrentModificationException on a RepNode, causing the RepNode to become unavailable. *[#24691]*
3. Corrected a problem in the configuration of the external tables example that caused the example to generate errors. The property name for specifying the name of the table in the `config.xml` file was incorrect — changed from "oracle.kv.table" to "oracle.kv.tableName". *[#24650]*
4. Fixed a problem in the implementation of the `show admins` command in the Administrator CLI that prevented it from authenticating correctly when used with a secure store. *[#24648]*
5. Improved the algorithm used to calculate the maximum number of threads used to perform concurrent table requests when the default is requested by specifying `TableIteratorOptions.maxConcurrentRequests` as `0`. This change affects the behavior of table multi-get operations, and table and index iterations. *[#24188]*
6. Fixed a security flaw in table-level access checking. Before this fix, it might have been possible for an authenticated client to access a table without permission if the user had permission to access other tables in the store.

Utility Changes

1. There were several changes made to the ping command to make it easier to use in scripts. *[#24407]*

   First, modified the `java -jar kvstore.jar ping` command to add a new flag, `-helper-hosts`, which is an alternative to the existing `-host` and `-port` flags. If multiple helper hosts are provided, the ping utility will have multiple nodes it can use to make an initial point of contact with the store, and will have a greater chance of success if some nodes of the store are unavailable. The new `-helper-hosts` flag can be specified as a comma-separated list of hosts and ports. For example:

   ```
   java -jar kvstore.jar ping -helper-hosts hst1:5000;hst2:5100
   ```

   Second, ping now generates an exit code which acts as a quick summary of the results. The exit code is returned both as a process exit code, if ping is called as a standalone utility, and as part of the JSON output. The exit code is meant to direct the caller to perform the appropriate follow-on action. The values, defined in the `Ping.ExitCode` enum, are:

   - 0 (EXIT_OK)

     All services in the store could be located and are in a known, good state (e.g. RUNNING).

   - 1 (EXIT_OPERATIONAL)

One or more services in the store could not be reached, or are in an unknown or not usable state. In this case the store should support all data operations across all shards, as well as all administrative operations, but may be in a state of degraded performance. Some action should be taken to find and fix the problem before part of the store becomes unavailable.

- 2 (EXIT_NO_ADMIN_QUORUM)

  The Admin Service replication group does not have quorum or is not available at all, and it's not possible to execute administrative operations which modify store configuration. The store supports all normal data operations despite the loss of admin quorum, but this state requires immediate attention to restore full store capabilities.

- 3 (EXIT_NO_SHARD_QUORUM)

  One or more of the shards does not have quorum and either cannot accept write requests, or is completely unavailable. This state requires immediate attention to restore store capabilities. This exit code takes precedence over EXIT_NO_ADMIN_QUORUM, so if this exit code is used it is possible that the admin capabilities are also reduced or unavailable.

- 100 (EXIT_USAGE)

  A usage error.

- 101 (EXIT_TOPOLOGY_FAILURE)

  Ping was unable to find a Topology in order to operate. This could be a store problem, a network problem, or it could be a usage problem with the parameters passed to Ping (e.g. the host and port or helper-hosts list are not part of a store).

- 102 (EXIT_UNEXPECTED)

  The utility has experienced an unexpected error.

Note that exit codes 1 through 3 may indicate a network connectivity issue that should be checked first before concluding that any services have a problem.

Finally, the following section has been added to Ping's JSON output, which can be requested using the `-json` flag, and appears when ping is called as a standalone command or as part of the Admin CLI. For example:

```
"operation" : "ping",
"return_code" : 5000,
"description" : "No errors found",
"exit_code" : 0
```

2. Some table commands in the Administrative CLI were deprecated in favor of using the execute command to perform DDL statements. The newly deprecated commands are: *[#23937]*

   - table
   - plan add-index
   - plan add-table
   - plan evolve-table
   - plan remove-index
   - plan remove-table

3. Some additional Admin CLI commands now support displaying output in JSON format. If `-json` is specified when starting the Admin CLI using `runadmin,` or is specified as an argument of a command, the execution result of the command will be displayed in a JSON format string containing standard NoSQL error messages. In addition to the `configure` command, support has been added for the following commands: *[#24346]*
   - plan deploy-zone
   - plan deploy-sn
   - pool create
   - pool join
   - plan deploy-admin
   - topology create
   - plan deploy-topology
   - show plan
   - verify configuration | upgrade | prerequisite
4. The admin CLI command `pool add` is now idempotent. Calling the command with a pool that already exists now returns without error. This change is intended to make the command easier to use in scripts. *[#24445]*
5. The `makebootconfig` command was modified to support the new `-dns-cachettl` flag. This flag specifies the number of seconds that replication nodes should cache host name to IP address mappings. A value of 0 means mappings should not be cached. A value of -1 means mappings should be cached indefinitely, which is the default. The value of the flag is used to set the "networkaddress.cache.ttl" and "networkaddress.cache.negative.ttl" security properties. Note that specifying a non-default value can have security implications. See the Java Networking Properties page for more details. *[#24087]*

Changes in 12cR1.3.4.7

The following changes were made in Oracle NoSQL Database 12cR1.3.4.7.

Topics

New Features

1. Added a new command to the CLI for reestablishing Admin service quorum that has been lost due to failure:

   ```
   repair-admin-quorum {-zn <id>|-znname <name>|-admin <id>}...
   ```

   The command repairs admin quorum by reducing membership of the admin group to the admins included in the specified zones and in the list of the specific admins. Reestablishing admin quorum is one of the first steps in performing disaster recovery. *[#23983]*

2. Added a new command to the Admin CLI to recover shard quorum lost due to failure by change zone types:

   ```
   plan failover { {-zn <zone-id>|-znname <zone-name>}
                   -type {primary|offline-secondary} }...
                 ...
   ```

   The command repairs shard quorum by changing the type of currently offline primary zones to secondary zones and, optionally, changing some secondary zones to primary zones. Failover is one of the final steps in performing a disaster recovery. *[#23546]*

3. Added the ability to change a zone type. Previously a zone could be created as either a PRIMARY or SECONDARY zone and once created the type could not be changed. With this release the new CLI command: `topology change-zone-type` allows the administrator to change a zone in a topology from PRIMARY to SECONDARY or from SECONDARY to PRIMARY. Once changed the topology can then be deployed as normal with the `plan deploy-topology` command. *[#24308]*

4. The Storage Node Agent (SNA) now supports a start-up mode that ensures its managed services are disabled before the SNA starts. If the `-disable-services` flag is specified on the SNA command line with the kvstore `start` command, the SNA's services will first be disabled before the SNA starts. If there is a failure disabling the services the SNA will not start. This ability is useful during disaster recovery to prevent unwanted services from starting.

The kvstore `disable-services` command has been removed. Instead use `stop -disable-services`. The `-disable-services` option can be used with the kvstore `start`, `stop`, and `restart` commands. *[#24337]*

5. Improved the `plan repair-topology` command. The command now can repair parameter differences for Admin and Replication Node services between the reference copy maintained by the Admin and the storage node configuration files or parameters in the running services. *[#23555]*
6. Added a new command to the CLI which will wait for replica nodes to reach a specified threshold of consistency with their masters. This command can be used when performing a switchover to wait for secondary nodes to catch up with their masters and to obtain information about progress towards reaching consistency. The new command:

   ```
   await-consistent -timeout <timeout-secs>
       [-zn <id> | -znname <name>]...
       [-replica-delay-threshold <time-millis>]
   ```

   will wait up to `timeout-secs` seconds for the replication delay of the replica nodes in the zones specified by `-zn` or `-znname` to be less than or equal to `time-millis` milliseconds. If no zones are specified then the command will wait on all replicas in the store. If `-replica-delay-threshold` is not specified the default value of 5000 milliseconds (5 seconds) is used.

   If any replicas do not reach the threshold within the timeout period the command will return with a list of those nodes along with their lag time, and information about whether the replica is catching up if known. *[#23982]*
7. Modified the output of the `ping` and `verify configuration` Admin CLI commands so that shard and zone status entries are named more clearly using non-overlapping categories. For shard status, the 'total' entry has been removed, the 'degraded' entry has been renamed to 'writable-degraded', and the 'noQuorum' entry has been renamed to 'read-only'. For zone status, the 'total' entry has been removed, and the 'offline' entry has been added. *[#24336]*
8. Modified the output of the `ping` and `verify configuration` Admin CLI commands to add an entry that provides an overview of the status of the Admin service nodes. Also modified the output of the `show admins` command to include information about the status and replication state of each admin. *[#23984]*
9. Modified the classes and interfaces in `oracle.kv.hadoop.table` and `oracle.kv.hadoop.hive.table` to support the Oracle NoSQL Database security model. Specifically, you can now run MapReduce jobs, Hive queries, and Big Data SQL queries against table data located in a store configured for secure access. In addition to the changes made in `oracle.kv.hadoop.table` and `oracle.kv.hadoop.hive.table` to support security, example code and detailed documentation has also been provided that explains and walks you through the details of executing a MapReduce job or Hive/Big Data SQL query against table data in a secure store. *[#23714], [#24107], [#24108]*
10. Modified the `oracle.kv.hadoop.hive.table.TableSerDe` and related classes to support executing Hive or Big Data SQL queries against data contained in tables whose schema may consist of complex (non-*primitive*) data types; that is, `Map`, `Record`, `Array` types. Additionally, documentation and examples have

been provided that walk you through the details of the data model and how it maps to the Hive and Big Data SQL data models. *[#24357], [#24358]*

11. Topology candidates with names that contain the dollar sign ('$') character are now reserved for use by the system, which may modify or remove them. The `topology create` and `topology clone` commands will print a warning if they are called to create topology candidates whose names contain the reserved character. Administrators should avoid creating or using topology candidates with names using dollar signs. *[#24386]*
12. The `plan start-services` and `plan stop-services` commands now support starting and stopping Admin services. Previously, these plans only operated on Replication Nodes (RNs). With this release the `-service` option accepts the name of an Admin service. In addition, if a zone is specified (with either the `-zn` or `-znname` flags) and the `-all-rns` is not specified, then all of the services, RNs and Admins, in that zone are affected. *[#24149]*
13. Some commands in the Admin CLI now supports displaying JSON format results. If `-json` is specified when starting the Admin CLI using `runadmin,` or is specified as an argument of a command, the execution result of the command will be displayed in a JSON format string containing standard NoSQL error messages. In this release, only the `configure` command has been changed to support JSON format result. *[#24444]*
14. The admin CLI commands `configure`, `topology create`, `pool join` and `plan deploy-admin` are now idempotent to support admin automation scripting. *[#24445]*
15. Added support for configuring replication nodes to use off-heap caches, which are disabled by default. An off-heap cache can be used to utilize large memories more efficiently than when using the same memory for the file system cache, while avoiding the Java GC overhead associated with large Java heaps. Please see Managing the Off-heap Cache for details. *[#24370]*

API Changes

1. The iterators `ParallelScanIterator` and `TableIterator` now support both `Direction.FORWARD` and `Direction.REVERSE.`

   Note that, due to the internal changes to support the new iterator options, the method `getStoreIteratorMetrics` on `oracle.kv.stats.KVStats` has been deprecated. This method returns an instance of `oracle.kv.stats.StoreIteratorMetrics` which has also been deprecated. All of the methods of the returned `StoreIteratorMetrics` object will return 0. *[#23583]*
2. Add Java Map based API for Record and Map field in `RecordValue` interface. Add Java `Iterable` and array based API for Array field in `RecordValue` interface. *[#24177]*:

   ```
   RecordValue putRecord(String fieldName, Map<String, ?> map)
   RecordValue putMap(String fieldName, Map<String, ?> map)
   RecordValue putArray(String name, Iterable<?> list)
   RecordValue putArray(String name, Object[] array)
   ```

3. The interfaces for the methods `oracle.kv.hadoop.table.TableInputFormat.setKVSecurity` and `oracle.kv.hadoop.hive.table.TableInputSplit.setKVStoreSecurity` have each

been changed to specify new information needed for security-related initialization; specifically,

- The name of the file that specifies the transport properties the client uses when connecting to the store (the *login file*).
- The PasswordCredentials containing the username and password the client presents to the store during authentication.
- The name of the file containing the public keys and/or certificates needed for authentication (the *trust file*). *[#23714], [#24107], [#24108]*

4. A new exception, `MetadataNotFoundException`, has been added to indicate that a client's metadata may be out of sync. It extends `FaultException` and can be caught by applications to trigger the need for a refresh of their metadata, and in particular, `Table` handles obtained via `TableAPI.getTable`.
5. Added bulk get APIs to fetch large numbers of keys and values efficiently based on discrete keys. The APIs can be used with both the Key/Value and Table data models. The following is a list of the new methods: *[#24285]*

   - New methods defined on `oracle.kv.KVStore`:

     ```
     ParallelScanIterator<KeyValueVersion>
     storeIterator(Iterator<Key>, int, KeyRange, Depth, Consistency,
     long, TimeUnit, StoreIteratorConfig)
     ParallelScanIterator<Key> storeKeysIterator(Iterator<Key>,
     int, KeyRange, Depth, Consistency, long, TimeUnit,
     StoreIteratorConfig)
     ParallelScanIterator<KeyValueVersion>
     storeIterator(List<Iterator<Key>>, int, KeyRange, Depth,
     Consistency, long, TimeUnit, StoreIteratorConfig)
     ParallelScanIterator<Key> storeKeysIterator(List<Iterator<Key>>,
     int, KeyRange, Depth, Consistency, long, TimeUnit,
     StoreIteratorConfig)
     ```

   - New methods defined on `oracle.kv.table.TableAPI`:

     ```
     TableIterator<Row> tableIterator(Iterator<PrimaryKey>,
     MultiRowOptions, TableIteratorOptions)
     TableIterator<PrimaryKey>
     tableKeysIterator(Iterator<PrimaryKey>, MultiRowOptions,
     TableIteratorOptions)
     TableIterator<Row> tableIterator(List<Iterator<PrimaryKey>>,
     MultiRowOptions, TableIteratorOptions)
     TableIterator<PrimaryKey>
     tableKeysIterator(List<Iterator<PrimaryKey>>, MultiRowOptions,
     TableIteratorOptions)
     ```

   Please review the javadoc associated with these new methods for details.

Bug and Performance Fixes

1. Fixed a problem where some plans would enter the INTERRUPTED state. This would happen when a plan needed to stop, or restart the master Admin which was running the plan. Once in the INTERRUPTED state, user action was required to have the plan resume execution. The fix allows the plan to automatically restart on

the new master Admin (either some other Admin, or the original Admin which has restarted).

Along with this fix, the Admin CLI has been updated so that it will wait for a plan even if the plan is restarted on an Admin which was not the Admin the CLI was originally connected to. This change affects the `plan` command's `-wait` flag as well as the `plan wait` command. *[#24535*]

2. Fixed a problem where `ArrayValue.set()` would result in the same behavior as `ArrayValue.add()`, mistakenly adding to the array rather that putting an entry at a specific index. *[#24326]*
3. Fixed a bug that a release 3.2 replication node cannot resolve the name of a role principal created by a new replication node running release 3.3 during an upgrade of a NoSQL cluster. This situation works properly when upgrading from the 3.2 release to the current release. *[#24502]*
4. Fixed a bug that caused the `makebootconfig` utility command to fail when it was called with a value for the `-capacity` flag that was greater than 1 when using a 32-bit Java virtual machine. *[#24550]*
5. Improved the performance of topology redistribution by batching transactional operations performed during partition migrations. *[#24529]*
6. Corrected problems where the table API implementation failed to perform case insensitive comparisons when comparing field names. *[#24553]*
7. Modified elasticity operations to provide improved store availability during an elasticity change by waiting for replicas to catch up with masters before counting them as contributing to the shard quorum. This change reduces the chance of a temporary loss of write availability during an elasticity change that moves replication nodes, particularly when the store is under heavy write loads. *[#24511]*

Utility Changes

1. The Admin CLI utility (`runadmin`) and the Admin CLI `connect store` command now support the following new flags to override the default configuration: *[#24101]*
   - `-timeout` to specify the store request timeout,
   - `-consistency` to specify the store request consistency, and
   - `-durability` to specify the store request durability.

Changes in 12cR1.3.3.4

The following changes were made in Oracle NoSQL Database 12cR1.3.3.4.

Topics

New Features

1. A number of new security features have been added.

   Users are now able to enforce table-level access checks through both the API and administrative command line interface (Admin CLI) by using the following new series of table-specific privileges:

Privilege	Description
READ_ANY_TABLE	Read from any table in kvstore
DELETE_ANY_TABLE	Delete data from any table in kvstore
INSERT_ANY_TABLE	Insert and update data to any tables in kvstore
READ_TABLE	Read from a specific table in kvstore
DELETE_TABLE	Delete data from a specific table in kvstore
INSERT_TABLE	Insert and update data to a specific table in kvstore
CREATE_ANY_TABLE	Create any table in kvstore
DROP_ANY_TABLE	Drop any table in kvstore
EVOLVE_ANY_TABLE	Evolve any table in kvstore
CREATE_ANY_INDEX	Create any index on any table in kvstore
DROP_ANY_INDEX	Drop any index on any table in kvstore
EVOLVE_TABLE	Evolve a specific table
CREATE_INDEX	Create index on a specific table
DROP_INDEX	Drop index on a specific table

   Users are now able to create new roles to group together privileges or other roles. This provides a way to grant a group of desired privileges to a user. New role management commands have been added to support create and drop roles, and to grant and revoke privileges or roles to and from other roles.

   Modifications to the data definition language (DDL) have been added to provide a declarative interface to all security operations. The language reference is in Security Guide. The language is accessible via the API as well as via the "execute" command in the Admin CLI.

   Passwords now have lifetimes and will expire when they have been in use beyond the specified lifetime. It is also possible now to explicitly expire a password when adding a new user or by altering the profile of an existing user. Users are required

to renew the expired password before they can log in to the store successfully. *[#23951]*

2. Admin service configuration has been enhanced to better match the use of primary and secondary zones. It is now possible for an Admin service to be a *primary* or a *secondary* Admin. This *Admin type* is analogous to zone type. A primary Admin can serve as a master or replica of the Admin shard, and vote in master elections. Secondary Admins can only serve as replicas and do not vote in master elections. All Admins created in earlier releases are primary Admins. With this release new Admins are created with the same type as their containing zone.

   Admin services created in a secondary zone by previous release will have the wrong type (primary). This mismatch will be reported as a *violation* from the `verify configuration` command. This condition can be remedied through the `plan repair-topology` command.

   In addition to the changes in Admin service type behavior, new rules have been put in place regarding Admin service deployment. In general it is recommended that Admin services follow the same rules as data nodes, specifically the number of Admin services in a zone should match the zone's replication factor. *[#23985], [#24182]*

3. The Command Line Interface (Admin CLI) now supports a *read only* mode if the Admin shard does not have quorum or if the master Admin is unreachable. In this read-only mode any commands which require the Admin to update persistent state, such as plan creation and execution, are disabled. However, most commands which provide status or configuration information will function. In addition to a notification in the CLI informing the user that the CLI is in this mode, the `show admins` command will also indicate that the CLI is connected read-only.

   Additional re-connect capabilities have been added to the Admin CLI to improve robustness of the CLI in the face of Admin node failures. *[#23943]*

API Changes

1. The methods `oracle.kv.table.TableAPI.execute` and `oracle.kv.table.TableAPI.executeSync` have been deprecated in favor of the new APIs, `oracle.kv.KVStore.execute` and `oracle.kv.KVStore.executeSync`. The motivation for the change is the introduction of new DDL statements which manage objects that are above the scope of a single table, such as users and roles. Likewise, the classes `oracle.kv.table.ExecutionFuture` and `oracle.kv.table.StatementResult` are deprecated in favor of `oracle.kv.ExecutionFuture` and `oracle.kv.StatementResult`. *[#23937]*

2. New methods have been added to `oracle.kv.table.RecordValue` to provide the ability to use JSON values for complex fields in a table. In the past, JSON could be used to specify the entire row, but not portions of a row. The following 6 JSON input methods for complex types in RecordValue are new: *[#24069]*

```
RecordValue putRecordAsJson(String fieldName, String jsonInput,
boolean exact);
RecordValue putRecordAsJson(String fieldName, InputStream
jsonInput, boolean exact);
RecordValue putArrayAsJson(String fieldName, String jsonInput,
boolean exact);
RecordValue putArrayAsJson(String fieldName, InputStream jsonInput,
boolean exact);
```

```
RecordValue putMapAsJson(String fieldName, String jsonInput,
boolean exact);
RecordValue putMapAsJson(String fieldName, InputStream
jsonInput,boolean exact);
```

3. New methods have been added to better support DDL use by stateless applications. `oracle.kv.ExecutionFuture.toByteArray()` returns a serialized version of the future that can later be passed to `oracle.kv.KVStore.getFuture(byte[])` to recreate a Future instance. *[#24228]*
4. There are changes to the way the results and status of DDL operations are reported in the API. Information about the execution of a DDL operation continues to be returned via `oracle.kv.StatementResult`. In past releases, both return values and execution status were returned via `StatementResult.getInfo()`, `StatementResult.getInfoAsJson()`. Now, results are return via a new method, `StatementResult.getResult()`, while status is returned via `StatementResult.getInfo()`, `getInfoAsJson`. *[#24277]*

Utility Changes

1. `plan deploy-admin` now deploys an Admin with the same type (primary or secondary) as the zone in which it resides. Previously the Admin was always created as a primary.
2. `plan repair-topology` will change an Admin's type so that the type matches the zone in which it resides.
3. `verify configuration` will report the following:
   - A *violation* for each Admin service whose type does not match the type of the zone in which it resides
   - A *violation* if the number of Admin services in a zone is less than the zone's rep factor
   - A *note* if the number of Admins services in a zone is greater than the zone's rep factor
4. The put command in the Admin CLI now has a new `-exact` flag. When true, the input json string or file specified to the command must contain values for all columns in the table, and cannot contain extraneous fields.
5. The Admin CLI now supports a new "show versions" command, which displays client and server version information. *[#24005]*
6. The Admin CLI now provides the ability to use the semicolon character as a command terminator to enter multiple commands, or to enter commands on multiple lines. *[#24119]*

   For example, the following command would have had to be typed in a single line:

   ```
   kv-> execute "create table users(name string, address string,
   primary key (name))"
   ```

   or

   ```
   kv-> execute "create table users \
       ->                 (name string, \
   ```

```
    ->                  address string, \
    ->                  primary key (name))"
```

but can now be entered this way:

```
kv-> execute "create table users
  ->                 (name string,
  ->                  address string,
  ->                  primary key (name))";
```

In addition, multiple commands can be entered as below, using a semicolon as a terminator.

```
kv-> show table -name users; get table -name users;
```

7. A new command line utility has been added that disables services associated with a storage node. You can use the new utility when starting a storage node whose services had configuration changes while the node was offline, to allow the configuration to be updated so that the services can be started with the proper configuration. *[#23988]*

   The new utility is invoked this way:

```
  java jar -kvstore.jar disable-services -root ROOT_DIRECTORY [-
config CONFIG_FILE]
```

8. Improvements have been made to the output of the Admin CLI `verify configuration` command, and both the Admin CLI and top level versions of the `ping` command: *[#23981]*

   - Output now uses JSON format if the new `-json` flag is specified
   - Output now includes summaries of the status of shards and zones in the store, and information about the replication status of non-master replication nodes
   - Output for the ping commands now includes information about admin services, to match the information provided by verify configuration

9. A new "verify" option has been added to the diagnostics command line utility. This option does a variety of health and configuration checks, such as:

   - configuration values are valid
   - inter-node clock skew is within the permissible maximum skew value
   - all nodes have supported Java versions
   - all nodes have correct network connectivity
   - all nodes have compatible NoSQL DB versions

```
java -jar kvstore.jar diagnostics verify -help

Usage: verify -checkLocal |
              -checkMulti
```

Bug and Performance Fixes

1. Changes have been made to transfer the RN master nodes on SN shutdown. When a SNA shuts down it checks if any of the RNs managed by the SNA is a master for its replication group. If an RN is a master, the SNA causes a master-transfer before shutting down the RN. The implementation performs the transfers one at a time so that each transfer can take into account the results of the previous one, to avoid overloading the target SNs. *[#22426]*
2. Modified the replication node and admin services to permit them to start up even if other members of a service's replication group were managed by storage nodes whose hostnames cannot be resolved via DNS. This change allows the store to continue to function, and in particular to support restarting services, if failures of storage nodes cause the nodes to have unresolvable DNS names. *[#23120]*
3. Modified the 'topology validate' CLI command. In addition to the 'violations' and 'notes' that are currently displayed by that command, it now also notes when the topology contains any zones that are empty; that is, the zones that contain no SNs. *[#23222]*
4. Currently the 'verify configuration' CLI command performs a component-by-component comparison of the store's current state or configuration against what is reflected in the store's Admin database, and displays any inconsistencies. Modified the command to also note when any empty zones exist in the store. With respect to the current the flags taken by the 'verify configuration' command, any changes related to empty zones should apply to all arguments except the optional `-sn` argument; in which case, the new behavior related to identifying empty data centers should not be executed. *[#23223]*
5. The CLI 'snapshot' commands currently allow one to collect a snapshot from each Admin and RN of every reachable SN in the store, or to remove one or all instances of previously collected snapshot data. These commands have been modified to take a '-zn' or '-znname' flag so that the command applies to all the SNs executing in the zone with the specified id or name. *[#23224]*
6. A bug has been fixed that could cause the Admin service to hang if the Admin web console is enabled, and is displaying log file output within the logtail pane, and an administrative command that updates Admin service configuration, such as plan deploy-admin, is running. *[23907]*
7. Clarified an error message that would occur when a field name was used instead of *elementof()* in a CHECK expression involving a map or array. *[#24055]*
8. Relaxed the DDL PRIMARY KEY expression, allowing the statement to redundantly specify single primary key fields as a SHARD keys as well. In a top-level (non-child) table the primary key is equivalent to the shard key unless otherwise specified. This change makes statements like this legal even though use of "SHARD" is redundant: *CREATE TABLE mytable(id INTEGER, PRIMARY KEY(SHARD(id)))*. *[#24105]*
9. Changes have been made to the request dispatcher to favor more rapid request failover on a node failure, reducing the possibility of a RequestTimeoutException. As part of this change, the default socket open timeouts have been reduced from 5 seconds to 3 seconds to permit a redispatch of requests within the request timeout period. *[#24152]*

Changes in 12cR1.3.2.15

The following changes were made in Oracle NoSQL Database 12cR1.3.2.15.

Topics

New Features

1. A declarative data definition language has been defined that allows declarative creation and management of tables and indexes. The language reference is in Data Definition Language for Tables. The language is accessible via APIs in Java as well as the new C driver. It can also be executed using the new "execute" command in the administrative command line interface (Admin CLI).
2. The Command Line Interface (Admin CLI) has a new "execute" command that can be used to execute data definition language (DDL) statements. The Java or C API is the preferred mechanism for defining tables, but if you wish to define and manage tables via the Admin CLI, the "execute" command is preferred over the previous "table" command.
3. A new C driver has been created to operate on tables and indexes in Oracle NoSQL Database. It is a separately downloadable product on the same page as this distribution. It is source code and must be compiled, along with its dependent libraries. Complete instructions are in its distribution.
4. The implementation of indexes on maps has been enhanced so that it is now possible to create an index on the key strings of a map as well as an index on the values of a map, making these indices much more useful for a variety of data models. These indexes result in multiple index entries for a given row, up to the number of map entries. Additional APIs were added to help use these indexes.
5. When using the Admin CLI interactively, you can now use backslash to enter a command on multiple lines. For example, you can now type:

   ```
   kv-> show events -type stat
    or
   kv-> show events \
   > -type stat
   ```

6. A new diagnostics command line utility is available to provide support for troubleshooting an Oracle NoSQL Database cluster. Currently, the "collect" functionality lets a user easily retrieve and package logfiles each node in the cluster for further analysis. Additional functionality will be rolled out over time. The new utility is invoked this way:

   ```
   java -jar kvstore.jar diagnostics

   diagnostics-> help
   Oracle NoSQL Database Diagnostic Utility Commands:
       setup
   ```

```
        collect
        exit
        help

    diagnostics->
```

API Changes

1. `oracle.kv.table.TableAPI.execute(String)` and `oracle.kv.table.TableAPI.executeSync(String)` were added to execute Data Definition Language (DDL) statements.
2. New interfaces, `oracle.kv.table.StatementResult` and `oracle.kv.table.ExecutionFuture` were added to handle results of the new statement execution methods on `TableAPI`.
3. `oracle.kv.table.MapValue.putNull(String)` and the constant, `oracle.kv.table.MapValue.ANONYMOUS`, were added in order to handle new the map indexes.
4. `oracle.kv.table.Index.createMapKeyFieldRange(String)` and `oracle.kv.table.Index.createMapValueFieldRange(String)` were added to create `FieldRange` instances for the new map indexes.
5. The method `oracle.kv.table.TableAPI.execute(List<TableOperation>, WriteOptions)` used to throw `oracle.kv.OperationExecutionException`. However, `OperationExecutionException` provides information that applies to the key value API in the `oracle.kv` package rather than the Table API in `oracle.kv.table`. The method has been changed to throw a new exception, `oracle.kv.table.TableOpExecutionException`, which provides information suitable for the Table API.

   Note that this is an incompatible API change. Applications which invoked `TableAPI.execute(List<TableOperation>, WriteOptions)` must be modified to handle `TableOpExecutionException` instead of `OperationExecutionException`.

Bug and Performance Fixes

1. The CLI verbose and hidden commands have been enhanced so they can be specified using on/off parameters in addition to the previous toggle type functionality. *[#23245]*
2. Modified the Admin CLI help to not display deprecated commands by default. A new `-includeDeprecated` flag was added for the CLI help command. If this flag is used, the deprecated commands are listed along with other commands. *[#23365]*
3. The `java -jar kvstore.jar load` utility did not provide proper feedback when there were errors in operation, and incorrectly returned "Load succeeded". Now it returns both a non-zero status code when an error has been found and an accurate status message. *[#23681]*
4. Fixed a security weakness that let arbitrary authenticated clients modify the store's topology by specifying an altered topology when propagating topology changes from one RN to another. A signature-based topology integrity check is employed to prevent it. *[#23709]*

5. A window existed where a secondary index could incorrectly lose entries if the store was expanded while updates to the underlying table were happening. This window has been closed. *[#23724]*
6. Fixed an issue that required users to reauthenticate to the store in order for changes made by granting or revoking roles for the current user to take effect. A real-time session update mechanism was introduced that allows existing login sessions to reflect role changes immediately without reauthentication. *[#23839]*
7. Fixed a bug that sometimes prevented re-execution of interrupted change-parameters plans, when Admin parameters were being changed. *[#23880]*
8. Fixed a problem where a client performing a parallel scan against a secure store without having permission to read the store would get a timeout exception rather than having the scan return no values. This problem could be encountered when using the `get kv -all` command with the administrative CLI:

   ```
   kv-> get kv -all
   Error handling command get kv -all: Failed to iterate records :
   Parallel storeIterator Request Queue take timed out.
   (12.1.3.1.3) Timeout: 5000ms
   ```

   If the scan was performed via the API using the `KVStore.storeIterator` or `storeKeysIterator` method overloadings that supply a `StoreIteratorConfig`, the problem could cause the method call to throw a `RequestTimeoutException`. In all of these cases, the iteration now returns no elements. *[#23881]*
9. Fixed an issue that would result in duplicate values returned from the Hadoop APIs when a complete primary key is specified. *[#23958]*
10. Fixed a bug where iteration over array indexes when the result set was larger than the batch size (default 100) could result in incorrect results or a hang. *[#23977]*
11. Fixed a slow memory leak in the handling of socket timeouts: The socket object associated with a failed socket connect operation continued to be referenced and could not be garbage collected. *[#24039]*

Changes in 12cR1.3.1.7

The following changes were made in Oracle NoSQL Database 12cR1.3.1.7.

Topics

New Features

1. Added a convenience method on the Table interface to construct a MultiRowOptions instance from a single list of table names and optional FieldRange. *[#23807]*
2. Added `FieldRange.getField()` to return the FieldDef instance that was used to construct the FieldRange object. *[#23908]*

Bug and Performance Fixes

1. Fixed a problem that caused index creation on an empty table to take as much as 10 minutes to complete. *[#23826]*
2. Fixed a performance regression in the CLI delete command. This fix may result in some delete commands being slower because the problem was an optimization for some delete cases, but in the general case delete will have the same performance it had in release 3.0.5. *[#23918]*
3. Removed unnecessary non-topology metadata broadcasts to replica rep nodes. Since only a master rep node can update the metadata for the shard there is no point in attempting to update a replica. With this change two new admin parameters have been added to control the metadata broadcast:
   - *broadcastMetadataDelay* specifies the broadcast metadata retry delay. This delay is the time between attempts to update RNs when trying to meet the metadata threshold. If not specified, the default delay is 10 seconds.
   - *broadcastMetadataThreshold* specifies the broadcast metadata threshold. The threshold is the percent of shards that must be successfully updated during a broadcast. If not specified, the default threshold is 20%.

   Note that this fix only affects the broadcast of table and security metadata. The topology broadcast parameters and behavior remain unchanged. *[#23362]*

Changes in 12cR1.3.1.5

The following changes were made in Oracle NoSQL Database 12cR1.3.1.5.

Topics

New Features

1. Added support for role-based authorization for both the API and the administrative CLI. Authorization is built on the authentication mechanism introduced in release 3.0. Each authenticated user of the KVStore can be granted roles which determine which APIs and CLI commands the user can access. This release only supports built-in roles, including the "read" and "write" roles for data access control, and the "sysadmin" and "dbadmin" roles for system and database operation control, respectively. More details of this feature are described in the Oracle NoSQL Database Security Guide.

   The KVStore logging system has also been enhanced to log security sensitive events. Two new logging levels named SEC_WARNING and SEC_INFO have been introduced. Messages logged at SEC_WARNING level will produce critical events in the CLI. High level security events like failed login attempts and unauthorized operations will be recorded as SEC_WARNING KVStore events. Execution of CLI commands which require sysadmin or dbadmin roles will be recorded as SEC_INFO events. Logging messages of all security sensitive events have "*timestamp* KVAuditInfo" as the prefix for the ease of grepping and filtering. *[#23423]*
2. The makebootconfig command has been enhanced to do validity checking of all parameters. The -force flag has been added to let the user manually override the default validity checks if desired. *[#23422]*

Bug and Performance Fixes

1. Fixed a bug that could cause the plan change-parameters command to fail in some cases, reporting a `NullPointerException` in the plan history. This could only come about in some situations where the parameter change requires restarting all nodes, but not all nodes are available. *[#22673]*
2. Modified heap size calculations for replication nodes to reduce heap sizes in some cases to permit using compressed object references.

   The supported Java environments can use compressed object references when the heap size is smaller than a certain size, currently between 25 and 32 GB depending on the implementation. Because the space savings provided by compressed references are substantial, using a larger heap size typically results in less usable space unless the heap size is at least 50% larger than the largest size that supports compressed references.

   The system now automatically reduces the replication node heap size if the value of the `memoryMB` storage node parameter produces a heap size that is too large to support compressed references but not large enough to provide more usable space. The heap size is not reduced if the application specifies a `javaMiscParams` replication node parameter that explicitly specifies either non-

compressed references or the maximum heap size. Note that, if the `memoryMB` parameter is not specified explicitly, or is set to zero, the system sets it to the amount of physical memory available on the host, and will reduce the heap size in the same way as when `memoryMB` is set explicitly to a non-zero value. *[#22695]*

3. The Version.getVersion method has been deprecated and removed from the public documentation. It will probably be removed entirely in a future release. The getVersion method returns an internal version number which is only meaningful relative to a particular shard. For that reason, it is not a safe to compare the method results for arbitrary versions, and so it has been removed. Applications can continue to use the equals method to compare versions for identity, but there is currently no official way to compare versions to determine which is newer. We could add that capability in the future if applications have a need for it. *[#23526]*
4. The plan add-index command has been changed to check for index creation status more promptly, which lets the command finish sooner. *[#23568]*
5. Added additional type validation when JSON input is used to construct Row and PrimaryKey instances. Previously it was possible to provide incorrect types that were silently cast to the correct type based on the table schema, possibly resulting in loss of information. *[#23765]*
6. Fixed a problem that could be encountered during the configuration of a NoSQL Database cluster hosted within an IBM J9 JVM environment. If the Storage Nodes were deployed without providing an explicit value for the optional memory_mb parameter, the available memory calculation for the Replication Node hosted on that SN was incorrect, which might cause an "OutOfMemoryError" for that replication node when the plan deploy-topology command was run. This could only happen on the J9 JVM, and has been fixed. *[#23737]*
7. Both Hadoop and Oracle External Tables employ independent processes to read data from a store in parallel. A change has been made to the way work is distributed among these processes. The new algorithm attempts to distribute work in a way that minimizes contention on a single replication node, while maximizing the amount of parallelism across the store. The number of shards, the replication factor, and the requested consistency, are input into the calculation. This change may also affect the number of Hadoop processes, or splits which are generated. In addition to the change in work distribution, both Hadoop and external table processes may use the Parallel Scan APIs, allowing multiple threads to operate in a single process. *[#23749]*

   > **Note:**
   >
   > The public method `oracle.kv.hadoop.KVInputFormatBase.setDirection(Direction)` has been deprecated since only `Direction.UNORDERED` is supported.

8. Fixed a problem where, if an index was created on an array, or field in an array, and the array itself is null in a Row, a `ClassCastException` could be thrown during index key extraction on a server node. `[#23757]`
9. Modified the Ant build script so that the default target, which builds the JAR files, works for the Community Edition. You can now use ant to build new versions of the JAR files after making modifications to the source code. *[#23764]*
10. If a client program made an unexpected, non-secure call to the trusted login service of an Storage Node, a flaw in the service implementation caused the

service to hang, causing future user logins to fail. This bug has been fixed. *[#23786]*

11. Fixed an issue with a runaway number of client threads when multiple index iterators are opened and closed in quick succession. Specifically, the iterator's close() method will now wait for its threads to exit before returning. This is the same behavior as the parallel scan iterators, though it is documented that it does not wait. The documentation has been corrected. *[#23797]*
12. Fixed a bug which could result in a deadlock situation when an index iterator is closed. As part of this fix the Javadoc for the `oracle.kv.KVStore` and `oracle.kv.table.TableAPI` interfaces have been updated to make it clear that the iterators returned by those interfaces can only be used safely by one thread at a time unless synchronized externally. *[#23799]*
13. In the course of upgrading a NoSQL cluster from R2.X to R3.x, when the first Storage Node upgrade occurs, the Replication Node it hosts will transition into "STARTING" status and will wait until more nodes are upgraded. If the user runs the "verify upgrade" command at this point, they would see the following, misleading error:

    ```
    kv-> verify upgrade
       Unknown Exception: class
    oracle.kv.impl.rep.admin.RepNodeAdminFaultException
       RepNode is not RUNNING, current status is STARTING (12.1.3.1.2)
       oracle.kv.impl.rep.admin.IllegalRepNodeServiceStateException:
       RepNode is not RUNNING, current status is STARTING
    ```

    This has been fixed. *[#23859]*
14. The following parameters used by the Berkeley DB JE storage engine used within NoSQL Database have been changed to reflect changes in the updated version of JE used by this release:
    - EnvironmentConfig.EVICTOR_MAX_THREADS which was previously set to 2 now uses the default JE value.
    - EnvironmentConfig.CHECKPOINTER_BYTES_INTERVAL has been changed from 200000000 to 500000000.
15. Fixed a performance issue where the stop utility command resulted in an incomplete checkpoint if the checkpoint took more than 10 seconds. The incomplete checkpoint could result in the replication node taking longer to restart than it would with a complete checkpoint. The *stop* command has also been modified to initiate checkpoints in parallel across all the replication nodes hosted by the storage node being shutdown, thus making the command run faster.
16. Fixed a problem that caused upgrading from earlier releases to a 3.0 release to fail in some cases, with the following stack trace in the Admin log. *[#23879]*

    ```
    com.sleepycat.persist.evolve.IncompatibleClassException: (JE 6.2.5)
    Changes to the fields or superclass were detected when evolving
    class:
    oracle.kv.impl.admin.plan.task.StopAdmin version: 1 to class:
    oracle.kv.impl.admin.plan.task.StopAdmin version:
    1 Error: A new higher version number must be assigned
    ---
    (Note that when upgrading an application in a replicated
    ```

```
environment, this exception may indicate that the Master was
mistakenly upgraded
before this Replica could be upgraded, and the solution is to
upgrade this Replica.)
    at
com.sleepycat.persist.impl.PersistCatalog.init(PersistCatalog.java:5
12)
    at
com.sleepycat.persist.impl.PersistCatalog.initAndRetry(PersistCatalo
g.java:268)
    at
com.sleepycat.persist.impl.PersistCatalog.<init>(PersistCatalog.java
:228)
    at com.sleepycat.persist.impl.Store.<init>(Store.java:202)
    at
com.sleepycat.persist.EntityStore.<init>(EntityStore.java:190)
    at oracle.kv.impl.admin.Admin.initEstore(Admin.java:2126)
```

Changes in 12cR1.3.0.14

The following changes were made in Oracle NoSQL Database 12cR1.3.0.14.

Topics

New Features

1. When using the Table API, it is now possible to create indices on fields in records, maps, and arrays. *[#23091]*
2. The integration of Oracle NoSQL Database with Oracle Coherence has been updated to support Coherence 12c (12.1.2). As of Coherence 12.1.2, cache configuration parameters are specified within a custom XML namespace and are processed by the NoSQL Database namespace handler at runtime. Though it's possible to use this updated module with Coherence version 3.7.1, we highly recommend that you upgrade Coherence to the latest version. Please see the javadoc for `oracle.kv.coherence` package for information on how to configure a NoSQL Database backed cache with Coherence 12.1.2, or the earlier Coherence 3.7.1. *[#23350]*
3. It is now possible to use Oracle External Tables to access Oracle NoSQL Database tables created with the Table API. In addition to the usual required properties, users need to specify the table name in the external table configuration file. Please see here for details. *[#23605]*
4. Added a new "size" option to the Admin CLI table command, to estimate the in-memory size of the given table. The results of the size command can be used as inputs when planning resource requirements for a store. *[#23444]*
5. Oracle Enterprise Manager can now monitor instances of Oracle NoSQL Database. For more information, please see Integrating Oracle Enterprise Manager with Oracle NoSQL Database in the Admin Guide.
6. It is now possible to access data written to an Oracle NoSQL Database via the Table API from within a Hadoop MapReduce job. In addition to the usual required properties, users now need to specify the table name in the command line used to initiate the MapReduce job. Please refer to the javadoc of the class, KVHOME/examples/hadoop/table/CountTableRows.java for additional details. *[#23714]*
7. It is now possible to execute Hive queries against data written to an Oracle NoSQL Database via the Table API. To employ this feature, one must create a Hive external table that is 'STORED BY' the new `oracle.kv.hadoop.hive.table.TableStorageHandler` class, with fields similar to the fields with which the Oracle NoSQL Database table was created; that is, same number and types. Additionally, the Hive TBLPROPERTIES must specify the store name, the helper host and ports, and the table name (which does not have to be equal to the Hive table name). *[#23714]*

Bug and Performance Fixes

1. Added more sanity checking and improved error messages for the securityconfig add/remove-security commands. *[#23311]*
2. Fixed a bug where specifying an invalid value for the Storage Node parameter, "mgmtClass" could cause a crash in the Admin service. *[#23227]*
3. Modified the `oracle.kv.RequestLimitConfig` constructor to improve bounds checking and correct problems with integer overflow. *[#23244]*
4. Fixed a bug that sometimes caused Admin parameters not to take effect until the Admin's hosting SNA was rebooted. *[#23429]*
5. Added a new attribute (String replicationState) in the RepNode MBean presented via JMX, to indicate the state of the RepNode's membership in its replication group. Typically the value will show "MASTER" or "REPLICA", but it can also report "DETACHED" or "UNKNOWN". This same value is reported via SNMP in the repNodeReplicationState object, as defined in nosql.mib. *[#23459]*
6. Modified the Durability, RequestLimitConfig, and Version classes to implement Serializable to permit serializing instances of `oracle.kv.KVStoreConfig`, which was already serializable. *[#23474]*
7. Fixed a bug where an operation using the Table API might see a SecondaryIntegrityException if there is a replication node failover while secondary index is populated. *[#23520]*
8. Prior to this release, it was not possible use the Load utility to create a new store from snapshot files that had been taken against a store with security enabled, or a store that was using the Table API. This has been fixed. The -security, -username, -load-admin, and -force flags were added to the load utility to use in this case. See the Administrator's Guide for more information. *[#23528]*
9. Fixed a bug where invoking the "history" command in the Admin CLI with the "-last" option and a value that is greater than the total number of commands executed in the store could result in a java.lang.ArrayIndexOutOfBoundsException. *[#23579]*
10. Fixed a bug where an Admin service might exit with the following exception. Before the fix, administrative functionality would seamlessly fail over to another admin service, but the process exit was unnecessary and would show up as an alertable event. *[#23580]*

    ```
    com.sleepycat.je.rep.UnknownMasterException:
    Transaction -XXX cannot execute write operations because this node
    is no longer a master
    ```

11. Fixed a bug in the Table API so that enum fields may have names that begin with an underscore.
12. In rare cases, when a store has been deployed with Storage Nodes with capacity > 1 and there are concurrent delete operations, table iteration operations, and a transfer of mastership roles in a shard, it could be possible for the iteration operation to incorrectly skip a value that should have been returned by the iterator. This has been fixed. *[#23608]*
13. Fixed a GC configuration issue that could cause the CMS phase of the Java GC to run repeatedly, consuming CPU resources on an otherwise idle RepNode. The fix changed the default JVM CMSInitiatingOccupancyFraction

from 77 to 80. Our testing indicates that this is a better configuration under a broad range of application access patterns. However, if you need to override this new configuration in some unusual circumstance, you can use the Admin's *change-policy* command and, if it's an existing store, the *plan change-parameter* command, as below: *[#23652]*

```
change-policy -params "javaMiscParams=-
XX:CMSInitiatingOccupancyFraction=77"
plan change-parameters -all-rns -params "javaMiscParams=-
XX:CMSInitiatingOccupancyFraction=77"
```

14. Fixed the following bugs which could occur in a store with security enabled:
    - The following error could be reported after a deploy-topology command which deploys a new replication node in a security enabled store: *[#23682]*

      ```
      Task 23/DeployNewRN on sn1(slc06tyu:5000) ended in state ERROR
      oracle.kv.impl.fault.RNUnavailableException: Security metadata
      database is not opened yet.
      ```

    - The following exception could be seen after an elasticity change in a security enabled store: *[#23703]*

      ```
      Insufficient access rights : client host: xx.xxx.xxx.xx:
      attempt to call
      RepNodeAdmin.updateMetadata(MetadataInfo,AuthContext,short)
      ```

    - The following problem might be logged after an elasticity change in a security enabled store: *[#23704]*

      ```
      ProcessMonitor: java.lang.NullPointerException
      ProcessMonitor: at
      oracle.kv.impl.api.rgstate.RepNodeState$ReqHandlerRef.resolve(Rep
      NodeState.java:649)
      ProcessMonitor: at
      oracle.kv.impl.api.rgstate.RepNodeState$ReqHandlerRef.get(RepNode
      State.java:709)
      ```

15. Fixed a bug where application requests might unnecessarily time out for a brief period of time directly after an elasticity change *[#23705]*

Packaging and Documentation Changes

1. The version of the Oracle Coherence library bundled with Oracle NoSQL Database has been upgraded to the more recent Coherence 12.1.2. This requires a change in the way cache configuration parameters are specified for the NoSQL Database backed cache.
2. New documentation has been added on how to use the Large Object API. See the index page, and "Oracle NoSQL Database Large Object API".

Changes in 12cR1.3.0.9

The following changes were made in Oracle NoSQL Database 12cR1.3.0.9.

Topics

New Features

1. Modified the administrative CLI to save its command line history to a file so that it is available after restart. If you want to disable this feature, the following Java property should be set while running runadmin:

   ```
   java -Doracle.kv.shell.jline.disable=true -jar KVHOME/kvstore.jar
   runadmin -host <hostname> -port <portname>
   ```

   The CLI attempts to save the history in a `KVHOME/.jlineoracle.kv.impl.admin.client.CommandShell.history` file, which is created and opened automatically. The default history saved is 500 lines. If the history file cannot be opened, it will fail silently and the CLI will run without saved history.

   The default history file path can be overridden by setting the `oracle.kv.shell.history.file=`*path* Java property.

   The default number of lines to save to the file can be modified by setting the `oracle.kv.shell.history.size=`*int_value* Java property. *[#22690]*

2. Modified the admin CLI `aggregate` command to provide subcommands for tables and key/value entries. The `aggregate table` subcommand performs simple data aggregation operations on numeric fields of a table, while the `aggregate kv` subcommand performs aggregation operations on keys. *[#23258]*

Bug and Performance Fixes

1. Modified the implementation of index iterators to use weak references so that the garbage collector can remove the resources associated with unused index iterators. *[#23306]*
2. Improved the handling of metadata propagation and other internal operations. *[#23355], [#23368], [#23385]*
3. Modified the external tables integration to distribute the concurrent processing load more evenly across processes. *[#23363]*
4. Fixed a problem where a failure during a partition migration performed during a topology redistribution for a store that has indexes resulted in `SecondaryIntegrityExceptions` being thrown when the migration was restarted. *[#23392]*
5. Removed the `FieldRange.setEndDate` method, in favor of the existing `setEnd` method. *[#23399]*

6. Modified schema evolution to prevent changes that could resurrect an old field using a different type. Such a change would cause old data to become unreadable by the current table. This fix prevents resurrection of a field name unless it exactly matches the previous definition. *[#23403]*

7. Fixed a problem with handling network timeouts that could result in `FaultException` being thrown from KVStore operations instead of RequestTimeout when a timeout occurs. Here's a sample stack trace: *[#23411]*

```
Caused by: oracle.kv.FaultException: Problem during unmarshalling
(12.1.2.1.24)
Fault class name: java.rmi.UnmarshalException
    at
oracle.kv.impl.api.RequestDispatcherImpl.faultIfWrite(RequestDispatc
herImpl.java:968)
    at
oracle.kv.impl.api.RequestDispatcherImpl.handleRemoteException(Reque
stDispatcherImpl.java:883)
    at
oracle.kv.impl.api.RequestDispatcherImpl.handleDispatchException(Req
uestDispatcherImpl.java:736)
    at
oracle.kv.impl.api.RequestDispatcherImpl.execute(RequestDispatcherIm
pl.java:572)
    at
oracle.kv.impl.api.RequestDispatcherImpl.execute(RequestDispatcherIm
pl.java:1031)
    at
oracle.kv.impl.api.KVStoreImpl.executeRequest(KVStoreImpl.java:1251)
    at
oracle.kv.impl.api.KVStoreImpl.putIfVersion(KVStoreImpl.java:990)
    at
oracle.kv.impl.api.KVStoreImpl.putIfVersion(KVStoreImpl.java:968)
    [...]
    ... 41 more
Caused by: java.rmi.UnmarshalException: Error unmarshaling return
header; nested exception is:
    java.net.SocketException: Socket closed
    at
sun.rmi.transport.StreamRemoteCall.executeCall(StreamRemoteCall.java
:228)
    at sun.rmi.server.UnicastRef.invoke(UnicastRef.java:161)
    at
java.rmi.server.RemoteObjectInvocationHandler.invokeRemoteMethod(Rem
oteObjectInvocationHandler.java:194)
    at
java.rmi.server.RemoteObjectInvocationHandler.invoke(RemoteObjectInv
ocationHandler.java:148)
    at com.sun.proxy.$Proxy21.execute(Unknown Source)
    at
oracle.kv.impl.api.RequestHandlerAPI.execute(RequestHandlerAPI.java:
94)
    at
oracle.kv.impl.api.RequestDispatcherImpl.execute(RequestDispatcherIm
pl.java:560)
```

```
    ... 46 more
Caused by: java.net.SocketException: Socket closed
    at
java.net.SocketOutputStream.socketWrite(SocketOutputStream.java:121)
    at
java.net.SocketOutputStream.write(SocketOutputStream.java:159)
    at
java.io.BufferedOutputStream.flushBuffer(BufferedOutputStream.java:8
2)
    at
java.io.BufferedOutputStream.flush(BufferedOutputStream.java:140)
    at
java.io.ObjectOutputStream$BlockDataOutputStream.flush(ObjectOutputS
tream.java:1822)
    at java.io.ObjectOutputStream.flush(ObjectOutputStream.java:718)
    at
sun.rmi.transport.StreamRemoteCall.releaseOutputStream(StreamRemoteC
all.java:114)
    at
sun.rmi.transport.StreamRemoteCall.executeCall(StreamRemoteCall.java
:212)
    ... 52 more
```

8. Modified table iteration to optimize performance when all matching entries fall within a single shard. *[#23412]*
9. Fixed an issue where the index scan iterator would fail to return records from a shard if there was a record that compared equal to a record from another shard. This situation occurred when there was more than one shard in a store and there were equivalent index entries for a given index in both shards. The symptom was index iteration returning fewer rows than expected. *[#23421]*
10. Added several interfaces to the table package:
    - `List<String>IndexKey.getFields()` to return the fields used to define the index.
    - `List<String>PrimaryKey.getFields()` to return the fields used to define the primary key.
    - `List<String>RecordValue.getFields()` to return the fields used to define the record, in declaration order.
    - `Map<String, FieldValue> MapValue.getFields()` to return an immutable view of the map.

    The related javadoc was also updated to indicate that the lists and maps returned from these, and similar interfaces, are immutable. *[#23433]*
11. The data Command Line Interface (CLI) has a method to input table rows from a file with a JSON representation. This input method had an issue where a blank line could cause an infinite loop in the input path. This has been fixed in a way that will result in silently skipping blank lines as well as comment lines (those whose first non-whitespace character is "#"). *[#23449]*
12. Fixed handling of null values in indexed fields and in IndexKey. Previously, a null value in an indexed field could cause a server side exception. During a put, null values in indexed fields will result in no index entries for indexes in which that field participates. Further, null values are not allowed in IndexKey instances.

IllegalArgumentException is thrown if an attempt is made to set a null value in an IndexKey. *[#23588]*

13. Modified the Admin Service to listen on all interfaces on a host. This change permits deployment of KVStore in heterogeneous network environments, where a hostname may be resolved to different IP addresses to make the best possible use of the available network hardware. *[#23524]*

Changes in 12cR1.3.0.5

The following changes were made in Oracle NoSQL Database 12cR1.3.0.5.

Topics

New Features

1. A new client interface has been added that includes a set of datatypes and a tabular data model using those types. The tabular data model is used to provide support for secondary indexes which are defined on fields in a table. The model is discussed in the Getting Started Guide for Tables

   Tables and indexes are defined using the administrative CLI and accessed via programmatic API. The data CLI has been enhanced to perform operations on tables and indexes as well. The API is documented in the Oracle NoSQL Database Javadoc, and is primarily in the *oracle.kv.table* package.

   It is possible to define tables that overlay data created with NoSQL DB Release 2 if that data was created using a conforming Avro schema. This overlay is required in order to create secondary indexes on conforming Release 2 data.

   The existing key/value interface remains available.

2. It is now possible to define secondary indexes for records. See the previous changelog entry about tables. Index entries for a given record have transactional consistency with their corresponding primary records. Index iteration operations are part of the new table API. Index scan operations allow applications to iterate over raw indexes in 3 ways -- forward order, reverse order, and unordered. It is possible to define exact match and range scans in indexes. Indexes can be on single fields or defined as composite indexes on multiple fields in a table.

3. Support for username/password authentication and secure network communications has been added. Existing applications that do not require this feature are not impacted except for a change to makebootconfig, which adds a new required argument (`-store-security`). Users that wish to use the new capabilities should be aware of the following areas of change:

   - The makebootconfig utility has a new required `-store-security` command-line argument, which is used to enable security for a new KVStore deployment. The -store-security flag is also used to enable a non-secure deployment, and is actually required in that case (-store-security none).
   - A new securityconfig tool has been added, which provides the ability to:
     - Create security configurations
     - Create/modified password stores
     - Enable security on a pre-existing, non-secure KVStore

  - Perform security maintenance tasks
- Additional methods on the KVStoreConfig class:
  - `setSecurityProperties()`
  - `getSecurityProperties()`
- An additional method on the KVStoreFactory class supporting authentication:
  - `KVStoreFactory.getStore(KVStoreConfig, LoginCredentials, ReauthenticateHandler)`

  This new method utilizes the new interfaces
  - `LoginCredentials`
  - `ReauthenticateHandler`

  As well as the new class `PasswordCredentials`.
- Additional methods on the KVStore interface:
  - `login()`
  - `logout()`
- New exception classes:
  - `UnauthorizedException`
  - `AuthenticationFailureException`
  - `AuthenticationRequiredException`

Users should also familiarize themselves with security property files, which are required when using a KVStore command-line utility program against a secure store, and which may also be useful when running an application against a secure store.

This feature is described in much greater depth in the Oracle NoSQL Database Security Guide, as well as in the Administrators Guide and product Javadoc.

4. The administrative CLI has been modified to use new terminology to refer to data centers. Data centers are now called *zones*. The new terminology is meant to clarify that these node groupings may not always coincide with physical data centers. A zone is a collection of nodes that have good network connectivity with each other and have some level of physical separation from nodes in other zones. That physical separation may mean that different zones are located in different physical data center buildings, but could also represent different floors, rooms, pods, or racks, depending on the particular deployment.

   Commands that contained the word "datacenter" have been deprecated, and are replaced with commands using the word "zone". The previous commands will continue to work in this release. New commands are:

   - `plan deploy-zone`
   - `plan remove-zone`
   - `show zones`

   Command flags that specify a zone have been changed to `-zn`, for a zone ID, and `-znname`, for a zone name. The earlier `-dc` and `-dcname` flags have been deprecated but will continue to work in this release. In addition, zone IDs can

now be specified using the "zn" prefix, with the earlier "dc" prefix still currently supported.

The administrative GUI has also been modified to use the new Zone terminology. *[#22878]*

5. There are now two types of zones. *Primary* zones contain electable nodes, which can serve as masters or replicas, and vote in master elections. All zones (or data centers) created in earlier releases are primary zones, and new zones are created as primary zones by default. *Secondary* zones contain nodes of the new *secondary node* type, which can only serve as replicas and do not vote in master elections. Secondary zones can be used to make a copy of the data available at a distant location, or to maintain an extra copy of the data to increase redundancy or read capacity. *[#22483]*

6. The show plan command now provides an estimated migration completion time. For example: *[#22183]*

```
Plan Deploy Topo (12)
State:                  RUNNING
Attempt number:         1
Started:                2014-01-14 17:35:09 UTC
Ended:                  2014-01-14 17:35:27 UTC
Total tasks:            27
 Successful:            12
 Incomplete:            15
Incomplete tasks
   3 partition migrations queued
   1 partition migrations running
   11 partition migrations succeeded, avg migration time = 550164
ms.
Estimated completion:  2014-01-14 19:57:37 UTC
```

7. A new read consistency option has been added for this release. `oracle.kv.Consistency.NONE_REQUIRED_NO_MASTER` can now be used to specify that the desired read operations must always be serviced by a replica, never the master. For read-heavy applications (ex. analytics), it may be desirable to isolate read requests so that they are performed only on replicas, never a master; reducing the load on the master. The preferred mechanism for achieving this sort of read isolation is the new secondary zone feature; which users are encouraged to employ for this purpose. But for cases where the use of secondary zones is not desired or impractical, `oracle.kv.Consistency.NONE_REQUIRED_NO_MASTER` can be used to achieve a similar effect, without the additional resources that secondary zones may require. *[#22338]*

8. The following methods have been added:

   - `KVStoreConfig.setReadZones()`
   - `KVStoreConfig.getReadZones()`

   These methods make it possible to require that read operations only be performed on nodes located in the specified zones.

9. The *show plans* command has been changed so that a range of plan history can be specified. With no arguments, *show plans* now displays only the ten most recently created plans, but new arguments can be used to select ranges by

creation time and by plan id. Issue the command "show plans -help" to see the complete set of options.

10. The makebootconfig utility has a new optional `-runadmin` command-line argument, which allows the SNA to force the start of a bootstrap admin even if the value of `-admin` is set to 0.

    The option `-port` of *plan deploy-admin* within the admin CLI has been changed to be able to control the start of the admin web service. No web service of the admin will be started if the `-port` is set to 0 in deploying.

    Users can also change the http port of an admin after deployment via *plan change-param* command of admin CLI to change the setting for whether an admin runs a web server. *[#22344]*

11. The *plan change-param* command has been changed to allow changing the parameters for a single admin service. *[#22244]*

12. NoSQL topology information is stored both in the Admin services and on Storage Nodes, and can become inconsistent if topology changing plans such as deploy-topology and migrate-sn are canceled before completion. Inconsistencies can be repaired by redeploying the target topology. In this release, a "plan repair-topology" command is also provided as an additional way of repairing topology inconsistencies. The verify configuration command now generates recommendations for when it may be beneficial to use repair-topology. *[#22753]*

Bug and Performance Fixes

1. The makebootconfig command now prints a message when it declines to overwrite existing configuration files. *[#23012]*

2. The "plan remove-admin" now permits removal of an Admin that is hosted by Storage Node that is not running. *[#23061]*

3. Fixed a bug that sometimes caused a duplication of the admin section in a Storage Node's config.xml file. As a result, the "plan change-parameters" command, when applied to an Admin service with this configuration irregularity, could unexpectedly have no effect. The bug could be provoked by attempting to deploy an Admin that is already deployed; but it could also happen when re-executing a failed "plan migrate-storagenode" command. *[#23152]*

4. Fixed a problem that caused the `topology redistribute` command to ignore storage directory settings when creating new replication nodes. *[#23161]*

5. Previously, when there was no activity during a RepNode's metrics-gathering period (the *statsInterval*), the previous period's metric values would be reported via JMX and SNMP. This behavior has changed so that the metrics are updated at every interval. *[#22842], [#22537]*

6. NoSQL DB automatically adjusts mastership identity so that master nodes are distributed across a store for optimal performance. Fixed a problem that prevented Master Balancing from being performed across multiple zones. *[#22857]*

7. Modified the LOB implementation to repeat calls to `InputStream.skip` as needed to position the input stream to the start location, so long as the calls return non-zero values. An `IllegalArgumentException` will be thrown if the calls do not advance the stream to the required start location.

Utility Changes

1. The administrative and data command line interfaces (CLI) have been merged into a single program. The usage of the merged CLI is compatible with most old usage but has additional options that allow it to work for administrative operations, data operations, or both. This change requires the use of kvstore.jar for data operations where in previous releases, the data CLI only required kvcli.jar, which depended on kvclient.jar.
2. The CLI has been enhanced with commands necessary to manage tables, indexes, security information, and zones.

Documentation Changes

1. With the introduction of the tabular data model and secondary indexes, a new Getting Started with the Table API guide has been added.
2. With the introduction of the new security features, a new Security Guide has been added.

Packaging Changes

1. The versions of the Avro and Jackson libraries bundled with Oracle NoSQL Database have been upgraded to the more recent Avro 1.7.6 and Jackson 1.9.3. These versions are compatible with the previous API versions.

Changes in 12cR1.2.1.57

The following changes were made in Oracle NoSQL Database 12cR1.2.1.57,

Topics

New Features

1. The new method `KVStore.appendLOB()` now permits appending to an existing LOB (Large Object). As part of this change, the method `PartialLOBException.isPartiallyDeleted()` has been deprecated in favor of the new method: `PartialLOBException.getPartialState()`. Please consult the javadoc associated with these new methods, as well as the updated doc for the interface `KVLargeObject`, for a detailed description of this new functionality.

   This release is backwards compatible with LOBs created in previous releases, with one exception: Only LOBs created in this, or a later, release support the append operation. Attempts to use the append operation on LOBs created in previous releases will result in the method throwing an `UnsupportedOperationException`.

   LOBs created in this release cannot be read or deleted by clients using earlier releases. Such operations will typically fail with a ConcurrentModificationException. Please ensure that all clients are updated to this release before creating new LOBs. *[#22876]*
2. GC log files for the Admin and RepNode services are now generated by default and placed in the `KVROOT/<storename>/log` directory (the standard location for all NoSQL related logging information). This default behavior only applies when using JDK release 1.7 or a later release, since gc log rotation is only supported in the more recent JDKs. The logging has minimal resource overheads. Having these log files readily available, conforms to deployment best practices for production java applications making it simpler to diagnose GC issues should the need arise. *[#22858]*

Bug and Performance Fixes

1. The heap requirement of the Admin service, when operating on a store that has undergone numerous changes, has been reduced. *[#21143]*
2. Fixed a bug in the Admin CLI "show plan -id <id>" command, which resulted in the omission of information about partition migration tasks from the plan history report. The command now correctly includes information about partition migrations. *[#22611]*
3. Reduce internal timeout values, associated with the network connection between a master and a replica, to permit faster master failover upon encountering a network hardware failure. *[#22861]*
4. An attempt to resume a failed put operation on a LOB larger than 3968K bytes could result in an incorrect ConcurrentModificationException in some circumstances. The bug has been fixed in this release. *[#22876]*

5. Changed the way plans are represented in the Admin's memory. Previously, there was no limit on the potential size of the in-memory representation of currently active and historical plans. With this fix, only active plans are kept in memory. *[#22963]*
6. Eliminated deadlocks in plan management in the Admin. *[#22992]*
7. A bug in the argument checking for the `StoreIteratorConfig` setter methods has been fixed. *[#23010]*
8. The makebootconfig command now prints a message when it declines to overwrite existing configuration files. *[#23012]*
9. The Replication Node configuration has been tuned to reduce CPU utilization when the Replication Node's cache is smaller than required, and cache eviction is taking place. *[#23026]*
10. The remove-admin command now permits removal of an Admin that is hosted by Storage Node that is not running. *[#23061]*
11. The show plans command could sometimes cause a crash in the Admin CLI because it would consume too much memory. This has been fixed. *[#23105]*

Changes in 12cR1.2.1.54

The following changes were made in Oracle NoSQL Database 12cR1.2.1.54.

Topics

- New Features
- Bug and Performance Fixes

New Features

1. Oracle NoSQL Database now offers a client only package. Oracle NoSQL Database Client Software Library is licensed pursuant to the Apache 2.0 License (Apache 2.0). The Apache License and third party notices for the Oracle NoSQL Database Client Software Library may be viewed at here.
2. A new overloading of the `KVStore.storeIterator()` and `KVStore.storeKeysIterator()` implements *Parallel Scans*. The other `storeIterator()` methods scan all shards and Replication Nodes in serial order. The new Parallel Scan methods allow the programmer to specify a number of client-side threads that are used to scan Replication Nodes in parallel. *[#22146]*

Bug and Performance Fixes

1. Improved error messages in the Data Command Line Interface (kvshell). For example, a put command with invalid inputs might have returned this error message in the past: *[#22791]*

   ```
   kvshell-> put -key /test -value ./emp.insert -file -json Employee
     Could not create JSON from input:
     Unable to serialize JsonNode
   ```

   but will now produce this more useful response:

   ```
   kvshell-> put -key /test -value ./emp.insert -file -json Employee
   Exception handling command put -key /test -value ./emp.insert -file
   -json Employee:
     Could not create JSON from input:
     Expected Avro type STRING but got JSON value: null in field
   Address of Employee
   ```

2. Fixed a bug when using the plan deploy-admin command. In some cases, if an Admin service encountered an error at start up, the process would become unresponsive. The correct behavior is for the process to shut down and be restarted by its owning Storage Node. *[#22908]*

Changes in 12cR1.2.1.25

The following changes were made in Oracle NoSQL Database 12cR1.2.1.25.

Topics

- Bug Fixes

Bug Fixes

1. If a Storage Node Agent process received a *master balancing* related remote request while shutting down, it could in rare instances throw an exception that would disable the master balancing function in the Storage Node Agent that initiated the request. This problem can be identified via the following (or similar) output in the log of the Storage Node Agent that initiated the request: *[#23419]*

```
2014-03-28 12:13:34.544 UTC SEVERE [sn2] MasterRebalanceThread
thread exiting due to exception.
null (12.1.2.1.24) java.lang.NullPointerException
    at
oracle.kv.impl.sna.StorageNodeAgentImpl$27.execute(StorageNodeAgentI
mpl.java:838)
    at
oracle.kv.impl.sna.StorageNodeAgentImpl$27.execute(StorageNodeAgentI
mpl.java:831)
    at
oracle.kv.impl.fault.ProcessFaultHandler.execute(ProcessFaultHandler
.java:119)
    at
oracle.kv.impl.sna.StorageNodeAgentImpl.getMDInfo(StorageNodeAgentIm
pl.java:829)
    at sun.reflect.GeneratedMethodAccessor12.invoke(Unknown Source)
    at
sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAcce
ssorImpl.java:43)
    at java.lang.reflect.Method.invoke(Method.java:606)
    ...
    at com.sun.proxy.$Proxy1.getMDInfo(Unknown Source)
    at
oracle.kv.impl.sna.StorageNodeAgentAPI.getMDInfo(StorageNodeAgentAPI
.java:492)
    at
oracle.kv.impl.sna.masterBalance.RebalanceThread.orderMSCNs(Rebalanc
eThread.java:580)
    at
oracle.kv.impl.sna.masterBalance.RebalanceThread.candidateTransfers(
RebalanceThread.java:485)
    at
oracle.kv.impl.sna.masterBalance.RebalanceThread.run(RebalanceThread
.java:301)
2014-03-28 12:13:34.546 UTC INFO [sn2] Master balance manager
shutdown
```

```
2014-03-28 12:13:34.546 UTC INFO [sn2] MasterRebalanceThread thread
exited.
```

Changes in 12cR1.2.1.24

The following changes were made in Oracle NoSQL Database 12cR1.2.1.24.

Topics

Bug and Performance Fixes

1. Under certain circumstances, a replication node which was on the verge of shutting down or in the midst of transitioning from master to replica state could experience this failure while cleaning up outstanding requests. Since the node would automatically restart, and the operation would be retried, the failure was transparent to the application, but could cause an unnecessary node failover. This has been fixed. *[#22152]*

   ```
   java.lang.IllegalStateException: Transaction 30 detected open
   cursors while aborting
       at com.sleepycat.je.txn.Txn.abortInternal(Txn.java:1190)
       at com.sleepycat.je.txn.Txn.abort(Txn.java:1100)
       at com.sleepycat.je.txn.Txn.abort(Txn.java:1073)
       at com.sleepycat.je.Transaction.abort(Transaction.java:207)
       at oracle.kv.impl.util.TxnUtil.abort(TxnUtil.java:80)
       at
   oracle.kv.impl.api.RequestHandlerImpl.executeInternal(RequestHandler
   Impl.java:469)
       at
   oracle.kv.impl.api.RequestHandlerImpl.access$300(RequestHandlerImpl.
   java:122)
       at
   oracle.kv.impl.api.RequestHandlerImpl$2.execute(RequestHandlerImpl.j
   ava:301)
       at
   oracle.kv.impl.api.RequestHandlerImpl$2.execute(RequestHandlerImpl.j
   ava:290)
       at
   oracle.kv.impl.fault.ProcessFaultHandler.execute(ProcessFaultHandler
   .java:135>
   ```

2. In past releases of NoSQL DB, a replication node which transitioned from master to replica state would have to close and reopen its database environment as part of the change in status. This transition has now been streamlined so that in the majority of cases, the database environment is not perturbed, the transition requires fewer resources, and the node is more available. *[#22627]*
3. The `plan deploy-topology` command has additional safeguards to increase the reliability of the topology rebalance and redistribute plans. When moving a replication node from one Storage Node to another, the command will now check that the Storage Nodes involved in the operation are up and running before any action is taken. *[#22850]*

4. Under certain circumstances it was possible for a replication node to use out of date master identity information when joining a shard. This could cause a delay if the targeted node was unavailable. This has been fixed. *[#22851]*
5. Under certain circumstances operations would end prematurely with oracle.kv.impl.fault.TTLFaultException. This exception is now handled internally by the server and client library and the operation is retried. If the fault condition continues the operation will eventually fail with a oracle.kv.RequestTimeoutException. *[#22860]*
6. Previously, there were cases where a replication node would require the transfer of a copy of the shard data in order to come up and join the shard, even though it was unnecessary. This has been fixed. *[#22782]*
7. When new storage nodes are added to an Oracle NoSQL DB deployment and a new topology is deployed, the store takes that opportunity to redistribute master roles for optimal performance. In some cases, the store might not notice the new storage nodes until other events, such as failovers or mastership changes had occurred, which caused a delay in master balancing. This has been fixed. *[#22888]*
8. The setting of the JE configuration parameter: je.evictor.criticalPercentage used by the store has been corrected. It used to be set to 105 and has been changed to 20. This new setting will provide better cache management behavior in cases where the data set size exceeds the optimal memory settings. *[#22899]*

Utility and Documentation Changes

1. A timestamp has been added to the output of the CLI "ping" command. *[#22859]*

Changes in 12cR1.2.1.19

The following changes were made in Oracle NoSQL Database 12cR1.2.1.19.

Topics

- Documentation Changes
- Bug and Performance Fixes

Documentation Changes

1. This release includes a new document, Oracle NoSQL Database Availability and Failover. It explains the general concepts and issues surrounding data availability when using Oracle NoSQL Database. The intended audiences for this document are system architects and developers. The new information can be found under the "For the Developer" section in the documentation index page.
2. Clarify the instructions for adding .avsc files to the classpath for the example on Avro bindings in <KVHOME>/examples/avro. Improve the error message when the .avsc files are not properly available.

Bug and Performance Fixes

1. Increased an internal parameter for lock timeouts from 500ms to 10 seconds. Since NoSQL DB ensures that data access is deadlock free, the small timeout values were unnecessary and could cause spurious errors in the face of transient network failures. *[#22583]*
2. Changing the store topology through the plan deploy-topology command could result in the following error if there was a transient network failure, or if the movement of the replication node took longer than a few seconds. Although the store state was still consistent, and the command could be manually retried, the command should be more resilient to communication glitches.

   ```
   ... [admin1] Task 2/RelocateRN ended in state ERROR with
   java.lang.RuntimeException: Time out while waiting for rg4-rn1 to
   come
   up on sn1 and become consistent with the master of the shard before
   deleting the RepNode from its old home on sn4 2/RelocateRN failed.

   java.lang.RuntimeException: Time out while waiting for rg4-rn1 to
   come
   up on sn1 and become consistent with the master of the shard before
   deleting the RepNode from its old home
   ```

   The command will now adjust waiting times and retry appropriately to ascertain whether the movement of a replication node has finished. *[#22596]*
3. Fixed a bug where a replication node would not restart automatically if the directory containing its data files was removed, or its data files were corrupted, but were later repaired. *[#22626]*
4. Added additional testing to reinforce the existing, correct behavior that a client directs write requests to the authoritative master in a segmented network split brain scenario. *[#22636]*

5. In some cases, the `java -jar kvstore.jar ping` command could generate spurious messages about components that are no longer legitimately within the store.

   ```
   Failed to connect to service commandService

   Connection refused to host: 10.32.17.12; nested exception is:
     java.net.ConnectException: Connection refused
           SNA at hostname:localhost registry port: 6000 has no
   available
           Admins or RNs registered.
   ```

   In particular, these messages could happen for bootstrap Admins on Storage Nodes that do not host deployed Admin Services. While the store was consistent, the error messages were confusing and have been removed. *[#22639]*

6. Fixed a small timing window in Replication Node master transfer that could incorrectly cause the transfer transaction catch up point to regress, when a master transfer is occurring under heavy application load. The result is that shard mastership can take too long a time or too short a time to transfer. If the transfer time is too short, the target master may not be optimally caught up, and a third member of the shard may detect this and throw an exception. *[#22658]*

7. Preemptively shut down and restart the replication node when a node transitions from master to replica, to reduce GC cost from refreshing the database environment. *[#22658]*

8. Made changes to the NoSQL client library to adapt to replication node failures more rapidly, by retrying or forwarding data requests when it detects that its original target is unavailable sooner. *[#22661]*

9. A NoSQL deployment could see this transient error when undergoing topology changes. Although the store remained consistent, the error messages were confusing and could incorrectly cause a `plan deploy-topology` command to fail. This has been corrected. *[#22678]*

   ```
    ... INFO [rg1-rn1] Failed pushing entire topology push to rg1-rn3
   updating from topo seq#: 0 to 1001 Problem:Update to topology seq#
   1001 failed ... oracle.kv.impl.fault.OperationFaultException:
     Update to topology seq# 1001 failed
        at
   oracle.kv.impl.rep.admin.RepNodeAdminImpl$6.execute(RepNodeAdminImpl
   .java:261)
        at
   oracle.kv.impl.fault.ProcessFaultHandler.execute(ProcessFaultHandler
   .java:169)
        at
   oracle.kv.impl.rep.admin.RepNodeAdminFaultHandler.execute(RepNodeAdm
   inFaultHandler.java:117)

    ...INFO [rg1-rn3] Topology update skipped. Current seq #: 1001
   Update seq #: 1001
   ```

10. Fixed a bug where a replication node which experiences an out of memory error did not restart automatically. *[#22679]*

11. Corrected the default calculation of available Storage Node memory when the Storage Node has been configured without a value for the bootstrap memory_mb parameter. In the past, the calculation was done using units of decimal megabytes, rather than MB, resulting in an overestimation of the appropriate replication node heap size. This default calculation is only used if the store has been configured without any bootstrap value for the memory_mb property, and the memory_mb storage node parameter has never been set. *[#22687]*
12. Update the Storage Node more quickly about the replica/master status of the replication nodes it hosts. The fix applies when executing the `plan deploy-topology` command on a store that contains Storage Nodes that have capacity values greater than 1, and can host multiple Replication Nodes. A delay in notifying the Storage Node of its replication nodes status can make the distribution of mastership responsibilities less optimal. *[#22689]*
13. Fixed a bug where the Admin service became unresponsive when executing a `plan deploy-topology` command. During this time, the admin service process appeared idle, only burning a second or two of CPU time once in a while and would not respond to new attempts to connect with the Admin CLI. The problem would likely only occur in large clusters with hundreds of components. *[#22694]*
14. Topology changes invoked by the `plan deploy-topology` which result in the movement of a replication node from one storage node to another are now more resilient to transient network failures. There are now more advance checks to ensure that the shard and storage nodes involved in the movement are available and ready to accept the change. In the advent of a network failure mid-move, the command is better at handling retries issued by the system administrator. *[#22722]*
15. Fixed a bug where application requests failed to be processed while the store is executing topology changes that require partition migration under heavy load. *[#22778]*
16. Adjust the default replication node garbage collection parameters to be more optimal, reducing CPU utilization in some cases. *[#22779]*
17. Reduce the time taken for a replica Replication Node to become up to date and available to handle application requests when it has fallen significantly behind due to downtime or to network communication failures. Previously, it exited and restarted the process before starting the catch up stage, but will now skip the restart. *[#22783]*
18. Fixed a bug where an internal queue in the Storage Node could fill up if its Replication Node repeatedly and unsuccessfully attempts to restart, as might happen when a resource is unavailable. In that case, the Storage Node was no longer able to automatically restart the replication node, and would have to be rebooted. *[#22786]*
19. Fixed a bug where a Replication Node that has been stopped due to repeated errors, perhaps due to a lack of resource, and then re-enabled with the "plan start-service" command, still did not restart. *[#22828]*
20. Fixed a bug where the following null pointer exception could happen for a restarting Replication Node. The problem was transient. *[#22830]*

```
INFO [sn1] rg2-rn2: ProcessMonitor: startProcess
   INFO [sn1] rg2-rn2: ProcessMonitor: stopProcess
   SEVERE [sn1] rg2-rn2: ProcessMonitor: Unexpected exception in
MonitorThread:
        java.lang.NullPointerExceptionjava.lang.NullPointerException
```

```
        at
oracle.kv.impl.sna.ProcessMonitor$MonitorThread.run(ProcessMonitor.j
ava:404)
```

21. Improve the client library's interpretation of UnknownHostException and ConnectIOException so that it more rapidly detects a network problem and updates its set of unavailable replication nodes. *[#22841]*

Changes in 12cR1.2.1.8

The following changes were made in Oracle NoSQL Database 12cR1.2.1.8.

Topics

New Features

This release includes support for upgrading the NoSQL DB software (client or server) without taking the store offline or without significant impact to ongoing operations. In addition, upgrades can be made incrementally, that is, it should not be necessary to update the software on every component at the same time. This support includes client and server code changes and new command line interface (CLI) commands. *[#22421]*

The new CLI commands provide the administrator tools to help with the upgrade process. Using these commands, the general upgrade procedure is:

1. Install the new software on a Storage Node running an admin service[1].
2. Install the new client and connect to the store.
3. Use the `verify prerequisite` command to verify that the entire store is at the proper software version to be upgraded (All 2.0 versions of NoSQL DB will qualify as prerequisites).
4. Use `show upgrade-order` to get an ordered list of nodes to upgrade.
5. Install the new software on the Storage Nodes (individually or in groups based on the ordered list).
6. Use the `verify upgrade` to monitor progress and verify that the upgrade was successful.

[1] In future releases this step will not be necessary

If the upgrade procedure is interrupted steps 4-6 can be repeated as necessary to complete the upgrade.

Bug and Performance Fixes

1. Unless configured specifically by the application, NoSQL DB specifies the -XX:ParallelGCThread jvm flag for each Replication Node process to indicate the number of garbage collector threads that the process should use. In the past, the algorithm in use generated a minimum value of 1 thread. After more testing, the minimum value has been raised to the min(4, <the number of cores on the node>). *[#22475]*

API Changes

1. The admin command line interface (CLI) provides the following new commands *[#22422]*:

   ```
   verify prerequisite [-silent] [-sn snX]*
   ```

   This command will verify that a set of storage nodes in the store meets the required prerequisites for upgrading to the current version and display the components which do not meet prerequisites or cannot be contacted. It will also check and report an illegal downgrade situation where the installed software is of a newer minor release than the current version. In this command the current version is the version of the software running the command line interface. If no storage nodes are specified, all of the nodes in the store will be checked.

   ```
   verify upgrade [-silent] [-sn snX]*
   ```

   This command will verify that a set of storage nodes in the store has been successfully upgraded to the current version and display the components which have not yet been upgraded or cannot be contacted. In this command the current version is the version of the software running the command line interface. If no storage nodes are specified, all of the nodes in the store will be checked.

   ```
   show upgrade-order
   ```

   This command will display the list of storage nodes in the order that they should be upgraded to maintain the store's availability. This command will display one or more storage nodes on a line. Multiple storage nodes on a line are separated by a space. If multiple storage nodes appear on a single line, then those nodes can be safely upgraded at the same time. When multiple nodes are upgraded at the same time, the upgrade must be completed on all nodes before the nodes next on the list can be upgraded.

   The `verify [-silent]` command has been deprecated and is replaced by `verify configuration [-silent]`. The `verify [-silent]` command will continue to work in this release.

Utility and Documentation Changes

1. In this release, the sample code provided by the utility class `WriteOperations` (located in the `examples` directory) now includes methods that perform write operations for *large objects* (or LOB, see KVLargeObject). The new utility methods added in this release will properly retry the associated LOB operation when a FaultException is encountered. Prior to this release, the `WriteOperations` utility only provided retry methods for objects that are not large objects. *[#21966]*
2. The number of JE lock tables used by Replication Nodes (controlled via the *je.lock.nLockTables* JE configuration parameter) has been increased from 1 to 97. This change helps improve performance of applications characterized by very high levels of concurrent updates, by reducing lock contention. *[#22373]*
3. The Administration CLI now permits the creation of multiple Datacenters. By choosing Datacenter replication factors so that each Datacenter holds less than

a quorum of replicas, this change makes it possible to create store layouts where the failure of a single Datacenter does not result in the loss of write availability for any shards in the store. In the current release, nodes in any Datacenter can participate in master elections and contribute to durability acknowledgments. As a consequence, master failover and durability acknowledgments will take longer if they involve datacenters that are separated by large distances. Future releases will provide greater flexibility in this area. *[#20905]*

Changes in 11gR2.2.0.39

The following changes were made in Oracle NoSQL Database 11gR2.2.0.39.

Topics

New Features

1. An integration with Oracle Coherence has been provided that allows Oracle NoSQL Database to be used as a cache for Oracle Coherence applications, also allowing applications to directly access cached data from Oracle NoSQL Database. This integration is a feature of the Enterprise Edition of the product and implemented as a new, independent jar file. It requires installation of the Oracle Coherence product as well. The feature is described in the Concepts Manual as well as the javadoc. *[#22291]*
2. The Enterprise Edition now has support for semantic technologies. Specifically, the Resource Description Framework (RDF), SPARQL query language, and a subset of the Web Ontology Language (OWL) are now supported. These capabilities are referred to as the RDF Graph feature of Oracle NoSQL Database. The RDF Graph feature provides a Java-based interface to store and query semantic data in Oracle NoSQL Database Enterprise Edition. The feature is described in the RDF Graph manual.

Bug Fixes

1. The preferred approach for setting NoSQL DB memory resources is to specify the memory_mb parameter for each SN when running the makebootconfig utility, and to let the system calculate the ideal Replication Node heap and cache sizes. However, it is possible to override the standard memory configurations by explicitly setting heap and cache sizes using the Replication Node javaMiscParams and cacheSize parameters. In past releases, setting the explicit values worked correctly when using the plan change-parameters command, but did not work correctly when using the change-policy command. This has been fixed, so that if desired, one can use the change-policy command for the javaMiscParams and cacheSize parameters to override the default memory allocation heuristics. *[#22097]*
2. A NoSQL DB deployment that executes on a node with no network available, as might happen when running a NoSQL DB demo or tutorial, would fail with this error:

   ```
   java.net.InetAddress.getLocalHost() returned loopback
   address:<hostname> and
     no suitable address associated with network interfaces.
   ```

   This has been fixed. *[#22252]*

API Changes

1. Prior to this release, if a write operation encountered an exception from the underlying persistent store indicating that the write completed on the shard's master but not necessarily on the desired combination of replicas within the specified time interval, then that exception would be swallowed and thus never propagated to the client. Originally, this behavior was considered desirable because not only is that exception rare (because of various preceding checks performed by the implementation), but swallowing the exception would keep the API simple by avoiding the introduction of an additional exception and/or additional communication at the API level. After further thought and discussion, the team concluded that clients should know when a write operation fails to complete because of such an exception. As a result, when such a condition occurs during a write operation, a `RequestTimeoutException` will now be propagated to the client; wrapping the original exception from the underlying persistent store as the cause. For additional information, including strategies one might employ when this exception is encountered, refer to the RequestTimeoutException javadoc.

   This has been fixed. *[#21210]*

2. A new parameter has been added which controls the display of records in exception and error messages. When `hideUserData` is set to true, as it is by default, error messages which are printed to the server side logs or are displayed via the show CLI commands replace any key/values with the string "[hidden]". To see the actual record content in errors, set the parameter to false. *[#22376]*

Utility and Documentation Changes

1. In previous releases, information about errors that occurred during NoSQL DB component start up as a result of a `plan deploy-topology` command would often be visible only within the NoSQL DB logs, which made installation troubleshooting difficult. In this release, such start up errors can now be seen via the Admin CLI `show plan -id <id>` command. *[#22101]*

2. The Storage Node Agent exposes MBeans on a non-default MBeanServer instance. In this release, the non-default MBeanServer now exposes the standard JVM platform MBeans as well as those relating only to Oracle NoSQL Database.

3. In both SNMP and JMX interfaces, the new totalRequests metric is now available. This metric counts the number of multi-operation sequences that occurred during the sampling period.

4. Prior to this release, the product was compiled and built against the 1.x version of Hadoop (CDH3). Thus, when employing a previous release, if one were to run the `examples.hadoop.CountMinorKeys` example against a cluster based on the 2.x version of Hadoop (CDH4), the MapReduce job initiated by that example would fail as a result of an `IncompatibleClassChangeError;` which is caused by an incompatibility introduced in `org.apache.hadoop.mapreduce.JobContext` between Hadoop 1.x and Hadoop 2.x. This failure occurs whether the example is compiled and built against Hadoop 1.x or Hadoop 2.x. Because the product's customer base almost exclusively uses Hadoop 2.x, this release will provide support for Hadoop 2.x instead of 1.x. Future releases may revisit support for both Hadoop version paths, but doing so will involve refactoring the codebase and its associated release artifacts, as well as substantial changes to the product's current build process.

   Support of Hadoop 2.x (CDH4) has been provided. *[#22157]*

5. The `java -jar kvstore.jar makebootconfig -mount` flag has been changed to `-storagedir`. The "plan change-mountpoints -path <storage directory>" command is deprecated in favor of "plan change-storagedir -storagedir <storage directory>". *[#21880]*
6. The concept of Storage Node capacity is better explained in the Administrator's Guide.
7. The Administrator's Guide has a revamped section on how to calculate the resources needed for operating a NoSQL DB deployment.

Changes in 11gR2.2.0.26

The following changes were made in Oracle NoSQL Database 11gR2.2.0.26.

Topics

New Features

1. This release adds the capability to remove an Admin service replica. If you have deployed more than one Admin, you can remove one of them using the following command:

   ```
   plan remove-admin -admin <adminId>
   ```

   You cannot remove the sole Admin if only one Admin instance is configured.

   For availability and durability reasons, it is highly recommended that you maintain at least three Admin instances at all times. For that reason, if you try to remove an Admin when the removal would result in there being fewer than three, the command will fail unless you give the *-force* flag.

   If you try to remove the Admin that is currently the master, mastership will transfer to another Admin. The plan will be interrupted, and subsequently can be re-executed on the new master Admin. To re-execute the interrupted plan, you would use this command:

   ```
   plan execute -id <planId>
   ```

2. The Admin CLI verify has an added check to verify that the Replication Nodes hosted on a single Storage Node have memory settings that fit within the Storage Node's memory budget. This guards against mistakes that may occur if the system administrator overrides defaults and manually sets Replication Node heap sizes. *[#21727]*
3. The Admin CLI verify command now labels any verification issues as violations or notes. Violations are of greater importance, and the system administrator should determine how to adjust the system to address the problem. Notes are warnings, and are of lesser importance. *[#21950]*

Bug fixes

1. Several corrections were made to latency statistics. These corrections apply to the service-side statistics in the Admin console, CLI `show perf` command, .perf files and .csv files, as well as the client-side statistics returned by KVStore.getStats. However, corrections to the 95% and 99% values do not apply to the client-side statistics, since these values do not appear in the client-side API. *[#21763]*
   - The definition of latency has been corrected for the "multi" operation requests (multiGet, multiDelete, execute, etc). These are labeled "multi" in the Op Type column where latency information is displayed. The previous definition

was "latency in milliseconds per *operation"* while the new definition is "latency in milliseconds per *request".* In other words, for a "multi" operation request, latency now applies to the entire request rather than to each operation. For "single" operation requests, the definition of latency has not changed.

- To go along with the change above, a new column containing the number of requests in the sample has been added to all latency information displays: `TotalReq`. This is also available for client-side statistics using the new `OperationMetrics.getTotalRequests` method. For "multi" operation requests, the total number of requests is normally smaller than the total number of operations (the `TotalOpscolumn)`. For "single" operation requests, the total number of requests and operations are equal.
- Improved the consistency of the values reported in each sample so that, for example, the minimum latency is always less than the maximum latency. However, note that statistics are collected without synchronization to avoid impacting performance, and for small sample sizes the values in a sample are not always accurate or self-consistent.
- Fixed a bug that caused the 95% and 99% values to show the maximum latency recorded (within 1000 ms), rather than the lowest 95% or 99% as intended. This bug only applied to the "multi" operation requests.
- Fixed a bug that caused the 95% and 99% values to sometimes mistakenly appear as -1. These values should only appear as -1 when there were no operations in the sample with a latency below 1000 ms.

2. Modified the Administration Process to allocate ports from within a port range if one is specified by the *-servicerange* argument to the *makebootconfig* utility. If the argument is not specified the Administration Process will use any available port. Please see the Configuring the Firewall for details regarding the configuration of ports used by Oracle NoSQL Database. *[#21962]*
3. Modified the replication node to handle the unlikely case that the locally stored topology is missing. A missing topology results in a java.lang.NullPointerException being thrown in the TopologyManager and will prevent the replication node from starting. *[#22015]*
4. Replication Node memory calculations are more robust for Storage Nodes that host multiple Replication Nodes. In previous releases, using the plan change-params command to reduce the capacity parameter for a Storage Node which hosts multiple Replication Nodes could result in an over aggressive increase in RN heap, which would make the Replication Nodes fail at start up. The problem would be fixed when a topology was rebalanced, but until that time, the Replication Nodes were unavailable. The default memory sizing calculation now factors in the number of RNs resident on a Storage Node, and adjusts RN heap sizes as Replication Nodes are relocated by the deploy-topology command. *[#21942]*
5. Fixed a bug that could cause a NullPointerException, such as the one below, during RN start-up. The exception would appear in the RN log and the RN would fail to start. The conditions under which this problem occurred include partition migration between shards along with multiple abnormal RN shutdowns. If this bug is encountered, it can be corrected by upgrading to the current release, and no data loss will occur. *[#22052]*

```
Exception in thread "main"
com.sleepycat.je.EnvironmentFailureException: (JE
5.0.XX) ...  last LSN=.../... LOG_INTEGRITY: Log information is
incorrect,
```

```
problem is likely persistent. Environment is invalid and must be
closed.
    at
com.sleepycat.je.recovery.RecoveryManager.traceAndThrowException(Rec
overyManager.java:2793)
    at
com.sleepycat.je.recovery.RecoveryManager.undoLNs(RecoveryManager.ja
va:1097)
    at
com.sleepycat.je.recovery.RecoveryManager.buildTree(RecoveryManager.
java:587)
    at
com.sleepycat.je.recovery.RecoveryManager.recover(RecoveryManager.ja
va:198)
    at
com.sleepycat.je.dbi.EnvironmentImpl.finishInit(EnvironmentImpl.java
:610)
    at
com.sleepycat.je.dbi.DbEnvPool.getEnvironment(DbEnvPool.java:208)
    at
com.sleepycat.je.Environment.makeEnvironmentImpl(Environment.java:24
6)
    at com.sleepycat.je.Environment.<init>(Environment.java:227)
    at com.sleepycat.je.Environment.<init>(Environment.java:170)
    ...
Caused by: java.lang.NullPointerException
    at
com.sleepycat.je.log.entry.LNLogEntry.postFetchInit(LNLogEntry.java:
406)
    at com.sleepycat.je.txn.TxnChain.<init>(TxnChain.java:133)
    at com.sleepycat.je.txn.TxnChain.<init>(TxnChain.java:84)
    at
com.sleepycat.je.recovery.RollbackTracker$RollbackPeriod.getChain(Ro
llbackTracker.java:1004)
    at
com.sleepycat.je.recovery.RollbackTracker$Scanner.rollback(RollbackT
racker.java:477)
    at
com.sleepycat.je.recovery.RecoveryManager.undoLNs(RecoveryManager.ja
va:1026)
    ... 10 more
```

6. Fixed a bug that causes excess memory to be used in the storage engine cache on an RN, which could result in poor performance as a result of cache eviction and additional I/O. The problem occurred only when the `KVStore.storeIterator` or `KVStore.storeKeysIterator` method was used. *[#21973]*

Performance and Other General Changes

1. The replicas in a shard now dynamically configure the JE property*RepParams.REPLAY_MAX_OPEN_DB_HANDLES* which controls the size of the cache used to hold database handles during replication. The cache size is determined dynamically based upon the number of partitions currently hosted by

the shard. This improved cache sizing can result in better write performance for shards hosting large numbers of partitions. *[#21967]*

2. The names of the client and server JAR files no longer include release version numbers. The files are now called:

   ```
   lib/kvstore.jar
   lib/kvclient.jar
   ```

   This change should reduce the amount of work needed to switch to a new release because the names of JAR files will no longer change between releases. Note that the name of the installation directory continues to include the release version number. *[#22034]*

3. A SEVERE level message is now logged and an admin alert is fired when the storage engine's average log cleaner (disk reclamation) backlog increases over time. An example of the message text is below.

   ```
   121215 13:48:57:480 SEVERE [...] Average cleaner backlog has grown
   from 0.0 to
   6.4. If the cleaner continues to be unable to make progress, the JE
   cache size
   and/or number of cleaner threads are probably too small. If this is
   not
   corrected, eventually all available disk space will be used.
   ```

   For more information on setting the cache size appropriately to avoid such problems, see "Determining the Per-Node Cache Size" in the Administrator's Guide. *[#21111]*

4. The storage engine's log cleaner will now delete files in the latter portion of the log, even when the application is not performing any write operations. Previously, files were prohibited from being deleted in the portion of the log after the last application write. When a log cleaner backlog was present (for example, when the cache had been configured too small, relative to the data set size and write rate), this could cause the cleaner to operate continuously without being able to delete files or make forward progress. *[#21069]*

5. NoSQL DB 2.0.23 introduced a performance regression over R1.2.23. The kvstore client library and Replication Node consumed a greater percentage of system CPU time. This regression has been fixed. *[#22096]*

Changes in 11gR2.2.0.23

The following changes were made in Oracle NoSQL Database 11gR2.2.0.23.

Topics

New Features

1. This release provides the ability to add storage nodes to the system after it has been deployed. The system will rebalance and redistribute the data onto the new nodes without stopping operations. See Determining your Store's Configuration in the Admin Guide, for details.
2. A new `oracle.kv.lob` package provides operations that can be used to read and write Large Objects (LOBs) such as audio and video files. As a general rule, any object larger than 1 MB is a good candidate for representation as a LOB. The LOB API permits access to large values without having to materialize the value in its entirety by providing streaming APIs for reading and writing these objects.
3. A C API has been added. The implementation uses Java JNI and requires a Java virtual machine to run on the client. It is available as a separate download.
4. Added a new remove-storagenode plan. This command will remove a storage node which is not hosting any NoSQL Database components from the system's topology. Two examples of when this might be useful are: *[#20530]*
   - A storage node was incorrectly configured, and cannot be deployed.
   - A storage node was once part of a NoSQL Database, but all components have been migrated from it using the migrate-storagenode command, and the storage node should be decommissioned.
5. Added the ability to specify additional physical configuration information about storage nodes including:
   - Capacity - the number of RepNodes the SN may host
   - Number of CPUs
   - Amount of memory to use
   - Specific directory paths (mount points) to use for RepNodes

   This information is used by the system to make more intelligent choices about resource allocation and consumption. The administration documentation discusses how these parameters are set and used. *[#20951]*
6. Added Avro support. The value of a kv pair can now be stored in Avro binary format. An Avro schema is defined for each type of data stored. The Avro schema is used to efficiently and compactly serialize the data, to guarantee that the data conforms to the schema, and to perform automatic evolution of the data as the schema changes over time. Bindings are supplied that allow representing Avro data as a POJO (Plain Old Java Object), a JSON object, or a generic Map-like

data structure. The `oracle.kv.avro` package is described in the Javadoc. The use of the Avro format is strongly recommended. NoSQL DB will leverage Avro in the future to provide additional features and capabilities. *[#21213]*

7. Added Avro support for the Hadoop `KVInputFormat` classes. A new `oracle.kv.hadoop.KVAvroInputFormat` class returns Avro `IndexedRecords` to the caller. When this class is used in conjunction with Oracle Loader for Hadoop, it is possible to read data directly from NoSQL Database using OLH without using an interim Map-Reduce job to store data in HDFS. *[#21157]*
8. Added a feature which allows Oracle Database External Tables to be used to access Oracle NoSQL Database records. There is more information in javadoc for the `oracle.kv.exttab` package and an "cookbook" example. *[#20981]*

Performance and other General Changes

1. The following new methods:
   - `KVStoreConfig.setOpenTimeout()`
   - `KVStoreConfig.getOpenTimeout()`
   - `KVStoreConfig.setReadTimeout()`
   - `KVStoreConfig.getReadTimeout()`

   have been added to allow clients to configure the socket timeouts used to make client requests. Please review the javadoc for details.

   R1 installations must ensure that the software on the storage nodes has been upgraded as described in the upgrade documentation accompanying this release before using the above APIs on the client. *[#20997]*
2. New service parameters have been added to control the backlog associated with sockets created by NoSQL Database. These are controllable for the Rep Node and Storage Nodes' Monitor, Admin, and Registry Handler interfaces. The parameters are `rnRHSOBacklog` (default 1024), `rnMonitorSOBacklog` (default 0), `rnAdminSOBacklog` (default 0), `rnAdminSOBacklog` (default 0), `snAdminSOBacklog` (default 0), `snMonitorSOBacklog` (default 0), and `snRegistrySOBacklog` (default 1024). *[#21322]*
3. Previously, calling `Key.isPrefix` with an argument containing a smaller major or minor path than the target Key object caused an IndexOutOfBoundsException in certain cases. This has been fixed.
4. The `KeyRange()` constructor now checks that the start `Key` is less than the end Key if both are specified, otherwise an `IllegalArgumentException` is thrown. `KeyRange` also has `toString()` and `fromString()` methods for encoding and decoding `KeyRange` instances, similar to the same methods in `Key`. *[#21470]*

Utility Changes

1. Many new commands have been added to the CLI. See Command Line Interface (CLI) Command Reference of the *Administrator's Guide* for details.
2. The Admin Console is now for monitoring only.
3. Administration CLI commands have been changed so that component ids match the ids used in the topology display. Previously Datacenters, Storage Nodes, Admin instances and Replication Nodes were identified only by number. For

example, the syntax to add Storage Node 17 to a Storage Node pool, or to show the parameters for a given Replication Node was: *[#21099]*

```
joinPool myStorageNodePool 17
show repnode-params 5,3
```

- Datacenters can now be expressed as # or dc#
- Admin instances can now be expressed as # or admin#
- Storage Nodes can now be expressed as # or sn#
- Replication Nodes can now be expressed as groupNum,nodeNum, or rgX-rnY

The commands shown above are still valid, but can also be expressed as:

```
joinPool myStorageNodePool sn17
show repnode-params rg5-rn3
```

Documentation, Installation and Integration

1. The javadoc for the `Key.createKey` methods has been improved to warn that List instances passed as parameters are owned by the Key object after calling the method. To avoid unpredictable results, they must not be modified. *[#20530]*

Changes in 11gR2.1.2.123

The following changes were made in Oracle NoSQL Database 11gR2.1.2.123.

Topics

Bug fixes

1. Previously, executing a change-repnode-params plan in order to change Replication Node parameters for a node other than the one running the Admin service would fail. This operation will now succeed. *[#20901]*
2. A deploy-storage-node plan which ran into problems when attempting to deploy a new storage node would leave the problematic SN in the store. This would require that the user either take manual action to remove the bad SN, or fix the problem and retry the plan. For convenience, the deploy-storage-node plan will now clean up if it runs into errors, and will not leave the failed SN behind. *[#20530]*

Performance and Other General Changes

1. The command line interface's `snapshot create` command has been made significantly faster. Previously, it could take minutes if executed on a store with a large amount of data. This should be reduced to seconds. *[#20772]*

Utility Changes

1. The two scripts for starting kvlite and executing control commands, `bin/run-kvlite.sh` and `bin/kvctl`, have been replaced by a `java -jar lib/kvstore-M.N.P.jar` command. This provides portability to all Java platforms, including Windows. The two scripts are deprecated, but will be supported for at least one release cycle.

   The translation from the old script commands to the new -jar commands is as follows:

Old script command	New -jar command
bin/run-kvlite.sh args...	java -jar lib/kvstore-M.N.P.jar kvlite args...
bin/kvctl command args...	java -jar lib/kvstore-M.N.P.jar command args...

   There are a few differences to be aware of between the old and new commands.

   - `nohup`, if desired, must be explicitly specified. In the `bin/kvctl` script, nohup was added automatically for the `start` and `restart` commands. To specify the equivalent command, use:

     ```
     nohup java -jar lib/kvstore-M.N.P.jar start args... > /dev/null
     < /dev/null 2<&1 &
     ```

- The logging configuration file for kvlite is now specified using standard Java syntax. Previously, the `examples/logging.properties` configuration file was added automatically when passing `-logging` to the `run-kvlite.sh` script. The new equivalent is:

  ```
  java -Djava.util.logging.config.file=examples/logging.properties
  -jar lib/kvstore-M.N.P.jar kvlite args...
  ```

- Previously, the `-host` argument defaulted to the local machine name (via the `` `hostname` `` command) when running the `kvctl` script. Now, for all control commands, no default hostname is used and the -host argument must be specified explicitly. This change was made for two reasons: 1) consistency, since the port and other arguments have no default value for control commands, and 2) safety, since specifying an explicit hostname guards against accidental errors.
- Previously, the `-host` argument defaulted to `localhost` when running the `run-kvlite.sh` script. Now, the default is the local machine name rather than (literally) `localhost`. Note that the kvlite command, unlike the control commands, has default values for all arguments. This is because the kvlite command is designed for ease-of-use during development on a single machine. kvlite should not be used in production or for performance testing.
- Previously, running `java -jar lib/kvstore-M.N.P.jar`, with or without arguments, printed the product version. Now, if no arguments are specified, a usage message is printed. To print the version, use the version command:

  ```
  java -jar lib/kvstore-M.N.P.jar version
  ```