MongoShake Script
수정(Comparison.py) / 동기화 후 정확도 체크

목차 :

MongoShake Script 수정

```
 # Include Option 추가

 # 3.7 이후 Pymongo  버전에 대한 업데이트

#!/usr/bin/env python

# -*- coding:utf-8 -*-

import pymongo
import time
import random
import sys
import getopt

# constant
COMPARISION_COUNT = "comparison_count"
COMPARISION_MODE = "comparisonMode"
EXCLUDE_DBS = "excludeDbs"
INCLUDE_DBS = "includeDbs"
EXCLUDE_COLLS = "excludeColls"
INCLUDE_COLLS = "includeDbs"
```

```
SAMPLE = "sample"
# we don't check collections and index here because sharding's
collection(`db.stats`) is splitted.
CheckList = {"objects": 1, "numExtents": 1, "ok": 1}
configure = {}

def log_info(message):
    print("INFO  [%s] %s " % (time.strftime('%Y-%m-%d %H:%M:%S'), message))

def log_error(message):
    print("ERROR [%s] %s " % (time.strftime('%Y-%m-%d %H:%M:%S'), message))

class MongoCluster:

    # pymongo connection
    conn = None

    # connection string
    url = ""

    def __init__(self, url):
        self.url = url

    def connect(self):
        self.conn = pymongo.MongoClient(self.url)

    def close(self):
        self.conn.close()


def filter_check(m):
    new_m = {}
    for k in CheckList:
        new_m[k] = m[k]
    return new_m

"""
    check meta data. include db.collection names and stats()
"""
def check(src, dst):
```

```
    #
    # check metadata
    #
    srcDbNames = src.conn.list_database_names()
    #The database_names is deprecated from 3.7 onwards and been replaced by
list_database_names()
    dstDbNames = dst.conn.list_database_names()
    if configure[INCLUDE_DBS] or configure[INCLUDE_COLLS]:
        include_flag = 1
        srcDbNames = [db for db in srcDbNames if db in configure[INCLUDE_DBS]]
        dstDbNames = [db for db in dstDbNames if db in configure[INCLUDE_DBS]]
    else :

        srcDbNames = [db for db in srcDbNames if db not in
configure[EXCLUDE_DBS]]
        dstDbNames = [db for db in dstDbNames if db not in
configure[EXCLUDE_DBS]]
    if len(srcDbNames) != len(dstDbNames):
        log_error("DIFF => database count not equals src[%s] !=
dst[%s].₩nsrc: %s₩ndst: %s" % (len(srcDbNames),

len(dstDbNames),

srcDbNames,

dstDbNames))
        return False
    else:
        log_info("EQUL => database count equals")

    # check database names and collections
    for db in srcDbNames:

        # if db in configure[EXCLUDE_DBS]:
        #     log_info("IGNR => ignore database [%s]" % db)
        #     continue

        if dstDbNames.count(db) == 0:
            log_error("DIFF => database [%s] only in srcDb" % (db))
            return False
```

```
        # db.stats() comparison
        srcDb = src.conn[db]
        dstDb = dst.conn[db]
        # srcStats = srcDb.command("dbstats")
        # dstStats = dstDb.command("dbstats")
        #
        # srcStats = filter_check(srcStats)
        # dstStats = filter_check(dstStats)
        #
        # if srcStats != dstStats:
        #     log_error("DIFF => database [%s] stats not equals src[%s],
dst[%s]" % (db, srcStats, dstStats))
        #     return False
        # else:
        #     log_info("EQUL => database [%s] stats equals" % db)

        # for collections in db
        srcColls = srcDb.list_collection_names()
        # The collection_names is deprecated from 3.7 onwards and been
replaced by list_collection_names()
        dstColls = dstDb.list_collection_names()
        #The collection_names is deprecated from 3.7 onwards and been replaced
by list_collection_names()
        if include_flag ==0 :
            srcColls = [coll for coll in srcColls if coll not in
configure[EXCLUDE_COLLS] and srcColls.count(coll) > 0]
            dstColls = [coll for coll in dstColls if coll not in
configure[EXCLUDE_COLLS] and dstColls.count(coll) > 0]
        else :
            srcColls = [coll for coll in srcColls if coll in
configure[INCLUDE_COLLS] and srcColls.count(coll) > 0]
            dstColls = [coll for coll in dstColls if coll in
configure[EXCLUDE_COLLS] and dstColls.count(coll) > 0]

        if len(srcColls) != len(dstColls):
            log_error("DIFF => database [%s] collections count not equals,
src[%s], dst[%s]" % (db, srcColls, dstColls))
            return False
        else:
            log_info("EQUL => database [%s] collections count equals" % (db))
```

```
        for coll in srcColls:
            if coll in configure[EXCLUDE_COLLS]:
                log_info("IGNR => ignore collection [%s]" % coll)
                continue

            if dstColls.count(coll) == 0:
                log_error("DIFF => collection only in source [%s]" % (coll))
                return False

            srcColl = srcDb[coll]
            dstColl = dstDb[coll]
            # comparison collection records number
            if srcColl.count() != dstColl.count():
                log_error("DIFF => collection [%s] record count not equals" %
(coll))
                return False
            else:
                log_info("EQUL => collection [%s] record count equals" %
(coll))

            # comparison collection index number
            src_index_length = len(srcColl.index_information())
            dst_index_length = len(dstColl.index_information())
            if src_index_length != dst_index_length:
                log_error("DIFF => collection [%s] index number not equals:
src[%r], dst[%r]" % (coll, src_index_length, dst_index_length))
                return False
            else:
                log_info("EQUL => collection [%s] index number equals" %
(coll))

            # check sample data
            if not data_comparison(srcColl, dstColl,
configure[COMPARISION_MODE]):
                log_error("DIFF => collection [%s] data comparison not
equals" % (coll))
                return False
            else:
                log_info("EQUL => collection [%s] data data comparison exactly
eauals" % (coll))
```

```
    return True


"""
    check sample data. comparison every entry
"""
def data_comparison(srcColl, dstColl, mode):
    if mode == "no":
        return True
    elif mode == "sample":
        # srcColl.count() mus::t equals to dstColl.count()
        count = configure[COMPARISION_COUNT] if configure[COMPARISION_COUNT]
<= srcColl.count() else srcColl.count()
    else: # all
        count = srcColl.count()

    if count == 0:
        return True

    rec_count = count
    batch = 16
    show_progress = (batch * 64)
    total = 0
    while count > 0:
        # sample a bounch of docs

        docs = srcColl.aggregate([{"$sample": {"size":batch}}])
        while docs.alive:
            doc = docs.next()
            migrated = dstColl.find_one(doc["_id"])
            # both origin and migrated bson is Map . so use ==
            if doc != migrated:
                log_error("DIFF => src_record[%s], dst_record[%s]" % (doc,
migrated))
                return False

        total += batch
        count -= batch

        if total % show_progress == 0:
```

```
            log_info("  ... process %d docs, %.2f %% !" % (total, total *
100.0 / rec_count))


    return True


def usage():
    print('|-------------------------------------------------------------------
-------------------------------------------------------------------------------
------------------------|')
    print("| Usage_with --exclude: ./comparison.py --src=localhost:27017/db? -
-dest=localhost:27018/db? --count=10000 (the sample number) --
excludeDbs=admin,local --excludeCollections=system.profile --
comparisonMode=sample/all/no (sample: comparison sample number, default; all:
comparison all data; no: only comparison outline without data)  |")
    print('|-------------------------------------------------------------------
-------------------------------------------------------------------------------
------------------------|')
    print("| Usage_with --include: ./comparison.py --src=localhost:27017/db? -
-dest=localhost:27018/db? --count=10000 (the sample number) --
includeDbs=test,itmes ( sample dbs )  --includeCollections=test.users --
comparisonMode=sample/all/no (sample: comparison sample number, default; all:
comparison all data; no: only comparison outline without data)  |")
    print('|-------------------------------------------------------------------
-------------------------------------------------------------------------------
------------------------|')
    print('| Like : ./comparison.py --src="localhost:3001" --
dest=localhost:3100  --count=1000  --excludeDbs=admin,local,mongoshake --
excludeCollections=system.profile --comparisonMode=sample  |')
    print('|-------------------------------------------------------------------
-------------------------------------------------------------------------------
------------------------|')
    exit(0)

if __name__ == "__main__":
    opts, args = getopt.getopt(sys.argv[1:], "hs:d:n:e:x:i:n", ["help",
"src=", "dest=", "count=", "excludeDbs=", "excludeCollections=",
"comparisonMode=","includeDbs=","includeCollections="])

    configure[SAMPLE] = True
```

```
    configure[EXCLUDE_DBS] = []
    configure[EXCLUDE_COLLS] = []
    configure[INCLUDE_DBS]=[]
    configure[INCLUDE_COLLS]=[]

    srcUrl, dstUrl = "", ""

    for key, value in opts:
        if key in ("-h", "--help"):
            usage()
        if key in ("-s", "--src"):
            srcUrl = value
        if key in ("-d", "--dest"):
            dstUrl = value
        if key in ("-n", "--count"):
            configure[COMPARISION_COUNT] = int(value)
        if key in ("-e", "--excludeDbs"):
            if key in ("-i","--includeDbs"):                                    #
To check if include option exist
                log_info("Cant use include / exclude in same time")             #
                exit(1)                                                         #
            configure[EXCLUDE_DBS] = value.split(",")
        elif key in ("-i","--includeDbs"):                                      #
            configure[INCLUDE_DBS] = value.split(",")                           #
        if key in ("-x", "--excludeCollections"):
            if key in ("-i","--includeDbs"):                                    #
                log_info("Cant use include / exclude in same time")             #
                exit(1)                                                         #
            configure[EXCLUDE_COLLS] = value.split(",")
        elif key in ("-n","--includeCollections"):                              #
            configure[INCLUDE_COLLS] = value.split(",")                         #

        if key in ("--comparisonMode"):
            print(value)
            if value != "all" and value != "no" and value != "sample":
                log_info("comparisonMode[%r] illegal" % (value))
                exit(1)
            configure[COMPARISION_MODE] = value
    if COMPARISION_MODE not in configure:
        configure[COMPARISION_MODE] = "sample"
```

```
    # params verify
    if len(srcUrl) == 0 or len(dstUrl) == 0:
        usage()

    # default count is 10000
    if configure.get(COMPARISION_COUNT) is None or
configure.get(COMPARISION_COUNT) <= 0:
        configure[COMPARISION_COUNT] = 10000

    # ignore databases
    configure[EXCLUDE_DBS] += ["admin", "local"]
    configure[EXCLUDE_COLLS] += ["system.profile"]

    # dump configuration
    log_info("Configuration [sample=%s, count=%d, excludeDbs=%s,
excludeColls=%s]" % (configure[SAMPLE], configure[COMPARISION_COUNT],
configure[EXCLUDE_DBS], configure[EXCLUDE_COLLS]))

    try :
        src, dst = MongoCluster(srcUrl), MongoCluster(dstUrl)
        print("[src = %s]" % srcUrl)
        print("[dst = %s]" % dstUrl)
        src.connect()
        dst.connect()
    #except (Exception, e):
    except Exception as e:
        print(e)
        log_error("create mongo connection failed %s|%s" % (srcUrl, dstUrl))
        exit()

    if check(src, dst):
        print("SUCCESS")
        exit(0)
    else:
        print("FAIL")
        exit(-1)

    src.close()
    dst.close()
```

2. 동기화 정확도 확인을 위한 테스트

새로 추가한 include 기능에 대한 테스트를 위해 새로운 db 와 collection 을 만들고 10 만개의 dummy data 를 추가했다.

```
mongos> use new_test
switched to db new_test
mongos> for(i=0;i<100000;i++) db.item.insertOne({"item":"item_num_"+i,"plz":"no_error"})
{
        "acknowledged" : true,
        "insertedId" : ObjectId("6355f3b9a56510eba562a78b")
}
```

과정 1 : 더미 데이터 생성

```
mongos> sh.enableSharding("new_test")
{
        "ok" : 1,
        "$clusterTime" : {
                "clusterTime" : Timestamp(1666577600, 2),
                "signature" : {
                        "hash" : BinData(0,"ae71Xj2wB04Jo1BkwjxFb9cLrg4="),
                        "keyId" : NumberLong("7153442991275769888")
                }
        },
        "operationTime" : Timestamp(1666577600, 2)
}
```

과정 2 : Enable database sharding

```
mongos> db.item.createIndex({_id:"hashed"})
{
        "raw" : {
                "shard2/172.32.21.124:27017,172.32.212.60:27017,172.32.30.9:27017" : {
                        "numIndexesBefore" : 1,
                        "numIndexesAfter" : 2,
                        "createdCollectionAutomatically" : false,
                        "commitQuorum" : "votingMembers",
                        "ok" : 1
                }
        },
        "ok" : 1,
        "$clusterTime" : {
                "clusterTime" : Timestamp(1666577532, 5),
                "signature" : {
                        "hash" : BinData(0,"a0N0kQYPBRRusO6BldzRKkpieLw="),
                        "keyId" : NumberLong("7153442991275769888")
                }
        },
        "operationTime" : Timestamp(1666577532, 5)
}
```

과정 3 : Index 생성

```
mongos> sh.shardCollection("new_test.item",{_id:"hashed"})
{
        "collectionsharded" : "new_test.item",
        "ok" : 1,
        "$clusterTime" : {
                "clusterTime" : Timestamp(1666577604, 27),
                "signature" : {
                        "hash" : BinData(0,"DSmmEZs3dYi7VsJRWzvJq8MQyrg="),
                        "keyId" : NumberLong("7153442991275769888")
                }
        },
        "operationTime" : Timestamp(1666577604, 23)
}
```

과정 4 : sharding

```
mongos> db.item.getShardDistribution()

Shard shard2 at shard2/172.32.21.124:27017,172.32.212.60:27017,172.32.30.9:27017
 data : 3.09MiB docs : 50001 chunks : 1
 estimated data per chunk : 3.09MiB
 estimated docs per chunk : 50001

Shard shard3 at shard3/172.32.162.46:27017,172.32.42.154:27017,172.32.51.237:27017
 data : 3.09MiB docs : 49999 chunks : 1
 estimated data per chunk : 3.09MiB
 estimated docs per chunk : 49999

Totals
 data : 6.18MiB docs : 100000 chunks : 2
 Shard shard2 contains 49.99% data, 50% docs in cluster, avg obj size on shard : 64B
 Shard shard3 contains 50% data, 49.99% docs in cluster, avg obj size on shard : 64B
```

과정 5 : 수 분뒤 sharding 이 완료된 모습을 확인 할 수 있다.

```
mongos> sh.disableBalancing("new_test.item")
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
```

과정 6 : Source db.collection 의 balancer 를 꺼주어야한다. sh.disableBalancing("_____ ")

```
mongos> show dbs
admin       0.000GB
config      0.004GB
mongoshake  0.000GB
new_test    0.019GB
test        0.235GB
```

← 현재 source cluster 상태

```
mongos> show dbs
admin   0.000GB
config  0.004GB
```

← 현재 dest cluter 상태

```
[root@ip-172-32-114-98 mongo-shake-v2.8.1]# ./collector.linux -conf=collector.conf -verbose 1
[2022/10/24 01:09:26 UTC] [INFO] log init succ. log.dir[] log.name[collector.log] log.level[info]
[2022/10/24 01:09:26 UTC] [INFO] MongoDB Version Source[5.0.13-11] Target[5.0.13-11]
[2022/10/24 01:09:26 UTC] [WARN]
_____________________________
\                            \        _          _____ |
 \                            \      / \___-=O'/|O'/__|
  \  MongoShake, Here we go !!  \_____\          / | /   )
  /                             /      '/-==__ _/__|/__=-|  -GM
 /        Alibaba Cloud       /        *           \ | |
/                            /                     (o)
----------------------------
```

error : must close balancer + oplog

```
ME5Ig=="}}]
[2022/10/24 04:58:38 UTC] [CRIT] shake exit, must close balancer in sharding + oplog
[2022/10/24 04:58:38 UTC] [CRIT] move chunk oplog found, must close balancer in sharding
```

: balancer 를 모두 꺼보았지만 실행되지 않았다. Source Cluster 에 있는 mongoshake dbs 를 삭제 해 주었더니 실행이 되었다.

include option Test 성공 .

```
[root@ip-172-32-114-98 mongo-shake-v2.8.1]# python3 ./comparison.py --src="mongodb://cluster_admin_user:124578@172.32.114.98:27018" --
sample
INFO  [2022-10-24 05:40:06] Configuration [sample=True, count=1000, include=['new_test'], includeColls=[]]
[src = mongodb://cluster_admin_user:124578@172.32.114.98:27018]
[dst = mongodb://cluster_admin_user:124578@13.213.44.47:27018]
INFO  [2022-10-24 05:40:07] EQUL => database count equals
INFO  [2022-10-24 05:40:07] EQUL => database [new_test] collections count equals
INFO  [2022-10-24 05:40:07] EQUL => collection [item] record count equals
INFO  [2022-10-24 05:40:07] EQUL => collection [item] index number equals
INFO  [2022-10-24 05:41:29] EQUL => collection [item] data data comparison exactly eauals
SUCCESS
[root@ip-172-32-114-98 mongo-shake-v2.8.1]#
```