MongoDB Pyspark 2023

github link

Dependency Management

- `org.mongodb.spark:mongo-spark-connector_2.12:10.2.1` is compiled against Scala 2.12, and supports **Spark 3.1.x** and above.

- `org.mongodb.spark:mongo-spark-connector_2.13:10.2.1` is compiled against Scala 2.13, and supports **Spark 3.2.x** and above.

- Java 8 or later

- MongoDB version 4.0 or later

document based on these versions:

mongo 7
spark 3.1.3
mongodb-spark-connector 10.2.1
scala 2.12
java 8

replication must be set in mongoDB for streaming:

convert the standalone server to a replica set
[document_link]

initialize step (create spark session):

```
my_spark = SparkSession \
   .builder \
   .appName("myApp") \
   .config("spark.mongodb.write.connection.uri",
"mongodb://localhost:27018/") \
   .config("spark.mongodb.read.connection.uri",
"mongodb://localhost:27018/") \
   .config("spark.jars.packages",
"org.mongodb.spark:mongo-spark-connector_2.12:10.2.1") \
   .getOrCreate()
```

set appName
config write/read connection uri (localhost:27018)
config mongo-spark connector jar package

**** use this session in all operation(my_spark)**

batch-mode read:

```
dataFrame = my_spark.read\
                .format("mongodb")\
                .option("database", "people")\
                .option("collection", "contacts")\
                .load()
# dataFrame.show(100)
dataFrame.printSchema()

dataFrame.show()

dataFrame.createOrReplaceTempView("temp")
my_spark.sql('select * from temp').show(100)
```

Required option:

set format(mongodb) →in older version set full package name
database name, collection name

batch-mode write:

insert sample dataframe in batch-mode

```
dataFrame = my_spark.createDataFrame(
   [('20', "amir", 50), ('21', "Gandalf", 1000), ('22', "Thorin", 195),
('23', "Balin", 178), ('4', "Kili", 77),
    ('5', "Dwalin", 169)], ["_id", "name", "age"])
dataFrame.write.format("mongodb") \
   .mode("append") \
   .option("database", "people") \
   .option("collection", "contacts") \
   .option("upsertDocument", "true") \
   .option('idFieldList', "name") \
   .save()
```

Required option:

set format(mongodb)
in older version set full package name
database name
collection name

mode:append or overwrite (in doc it specify for upcoming version)

stream-mode read:

this code create stream read data from mongo and print in console

```
# define the schema of the source collection
readSchema = (StructType()
 .add('_id', StringType())
 .add('name', StringType())
 .add('age', IntegerType())
)
# define a streaming query
dataStreamWriter = (my_spark.readStream
 .format("mongodb")
 .option('spark.mongodb.database','people')
 .option('spark.mongodb.collection', 'contacts')
 .option ("forceDeleteTempCheckpointLocation", "true")
 .option("checkpointLocation", "/tmp")
 .schema(readSchema)
 .load()
 # manipulate your streaming data
 .writeStream
 .format("console")
 .option("checkpointLocation", "/tmp")
 .trigger(continuous="3 second")
 .outputMode("append")
)
# run the query
query = dataStreamWriter.start().awaitTermination()
```

Required option:

To create a structured stream from mongo we must specify schema as defined in code.
set format(mongodb) → in older version set full package name
database name
collection name

** checkpointLocation used for file that store metadata of check point for Fault Tolerance
** use awaitTermination() to wait until end of process

stream-mode write:

write csv file to mongo with stream write:

```
csv_schema = (StructType()
             .add('name', StringType())
             .add('country', StringType())
             .add('email', IntegerType())
             )

dataStreamWriter = (my_spark.readStream
                   .format("csv")
                   .option("header", "true")
                   .schema(csv_schema)

.load("/home/amirhosein/PycharmProjects/mongoPyspark4/csv_data/")
                   # manipulate your streaming data
                   .writeStream
                   .format("mongodb")
                   .option("checkpointLocation", "/tmp/pyspark7/")
                   .option("forceDeleteTempCheckpointLocation", "true")
                   .option("spark.mongodb.database", 'people')
                   .option("spark.mongodb.collection", 'contacts3')
                   .outputMode("append")
                   )

dataStreamWriter.start().awaitTermination()
```

Required option:

set format(mongodb)
in older version set full package name
database name
collection name

set csv schema
checkpointLocation used for file that store metadata of check point for Fault Tolerance
use awaitTermination() to wait until end of process

note: If it needed to insert more than one time, delete checkpoint file