Test cases

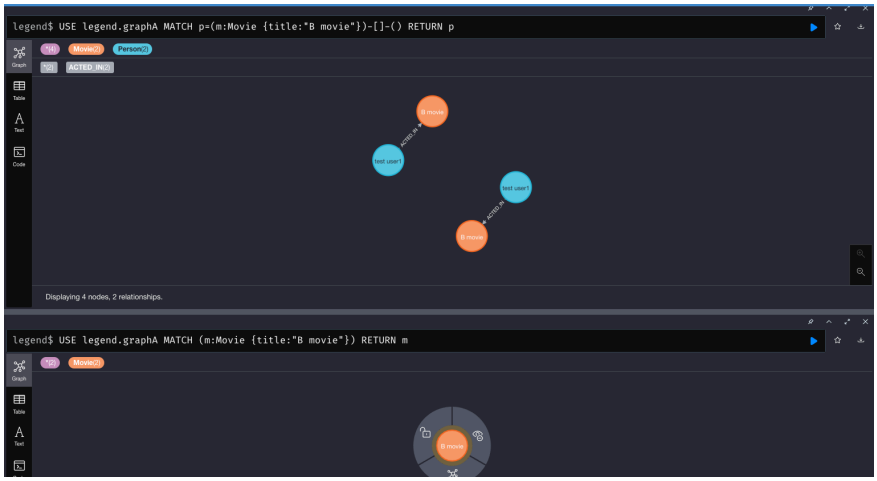
Monday, August 9, 2021 10:44 AM

1. Loading the data to a single data base
    a. using load csv using file.
        i. Question comes if we have two routing fabric servers which import folder we have to use.
            1) Need to load data to both.
            2) If the call is going to the other fabric server where files are not there, will get error "Couldn't load the external resource at: file:/home/ec2-user/neo4j-enterprise-4.3.2/import/orders.csv"
        ii. Need to use cypher subquery as csv files are on fabric and query need to be executed on different server.
        iii. Load csv will have one query for one record
            1) {@@row: {ShipName: 'Split Rail Beer & Ale', RequiredDate: '1997-12-25', Discount: '0.2', ShipCity: 'Lander', ShippedDate: '1997-12-02', Quantity: '20', ProductID: '69', ShipVia: '2', CustomerID: 'SPLIR', ShipPostalCode: '82520', OrderID: '10756', ShipRegion: 'WY', OrderDate: '1997-11-27', UnitPrice: '36', ShipAddress: 'P.O. Box 555', ShipCountry: 'USA', EmployeeID: '8', Freight: '73.21'}} - runtime=null - {}
    b. using https
        i. It is straight forward.
            1) What is happening in fabric server?
            2) Wha tis happening in write server?
            3) The same query
    c. using batch:
        i. Batch can be done in two ways using
2. Transaction with multiple writes to the database:
    a. Scenario 1: USE statement in each query
        i. In the transaction when each query has mentioned with USE database, query run successfully and created all node.
        ii. Browser behavior:
            1) When using with the fabric browser:
                a) Created Relations are not automatically open when we double click on the node. It made me to think relation not existing.

b) When we go to the primary data base and see, relations are existing.

c) On primary database if we run the return path, it is returning the relations too.

d)



b. Scenario 2 : USE statement once

i. If USE db is not mentioned, it will try to write to the local database.

ii. Error : 2021-08-09 15:14:50,178 [ERROR] {code: Neo.ClientError.Statement.AccessMode} {message: Writing to more than one database per transaction is not allowed. Attempted write to Local.

iii. As the error is at transaction level, even the first query did not created the node. Might be it got rollbacked.

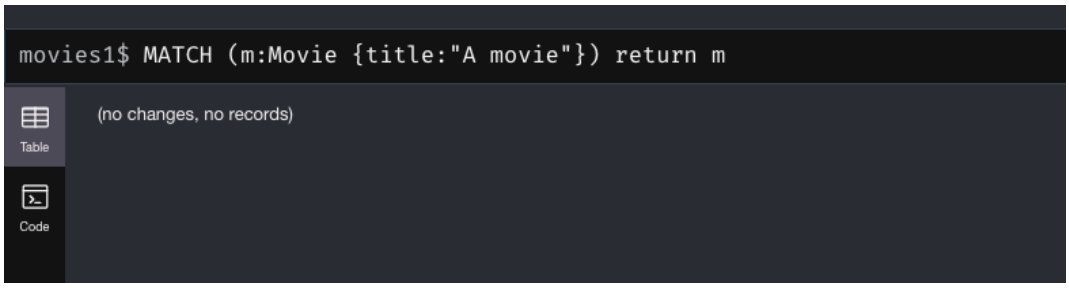
3. Transaction boundary testing:

a. Running 3 write queries and pause the application. Observe any commit happened or not?

b. Result: ran as expected. As commit was not happened at that instant nodes were not showing up.

i.

ii.

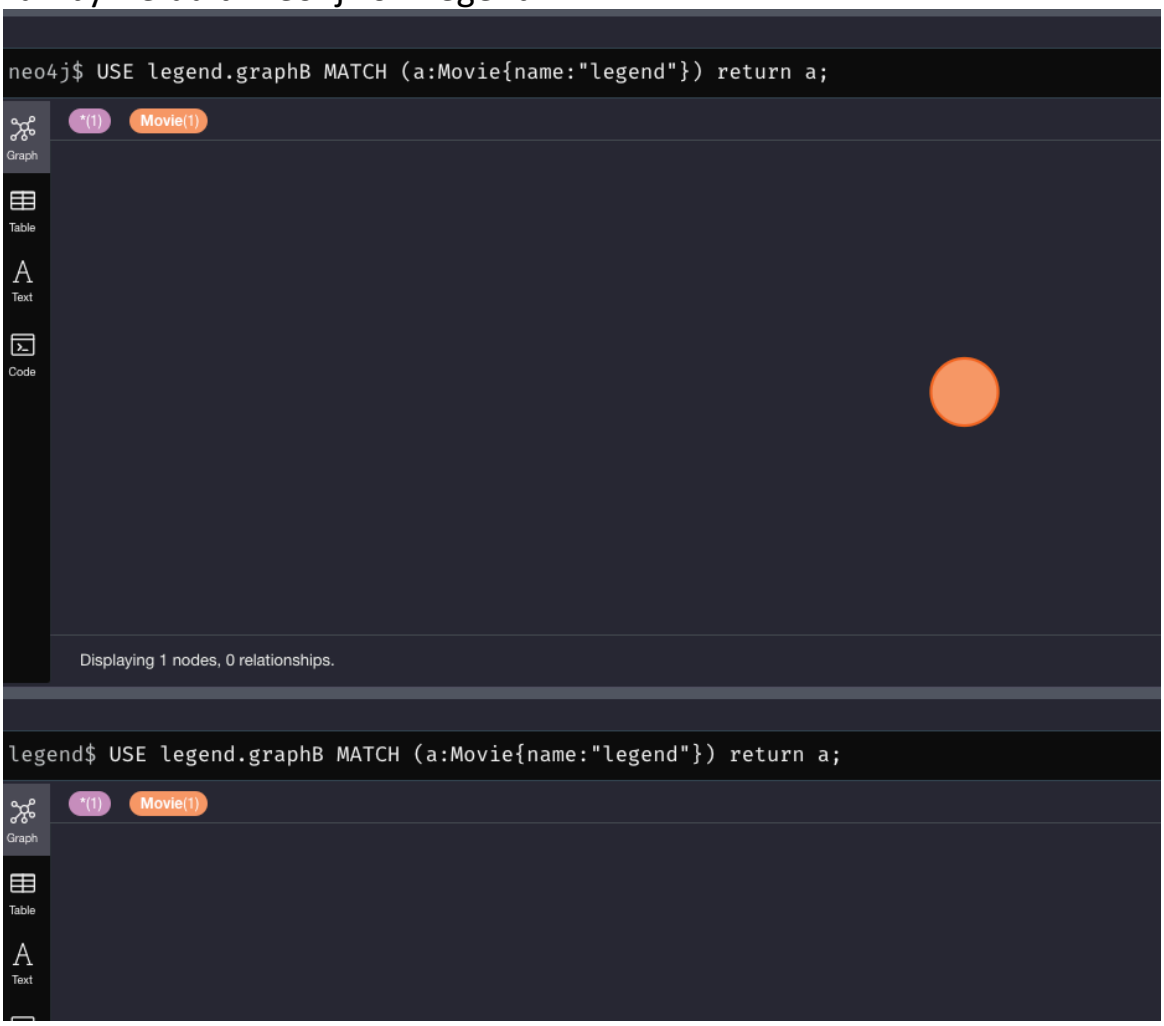


iii. After running all steps: Node appered. So transaction boundary is respected.

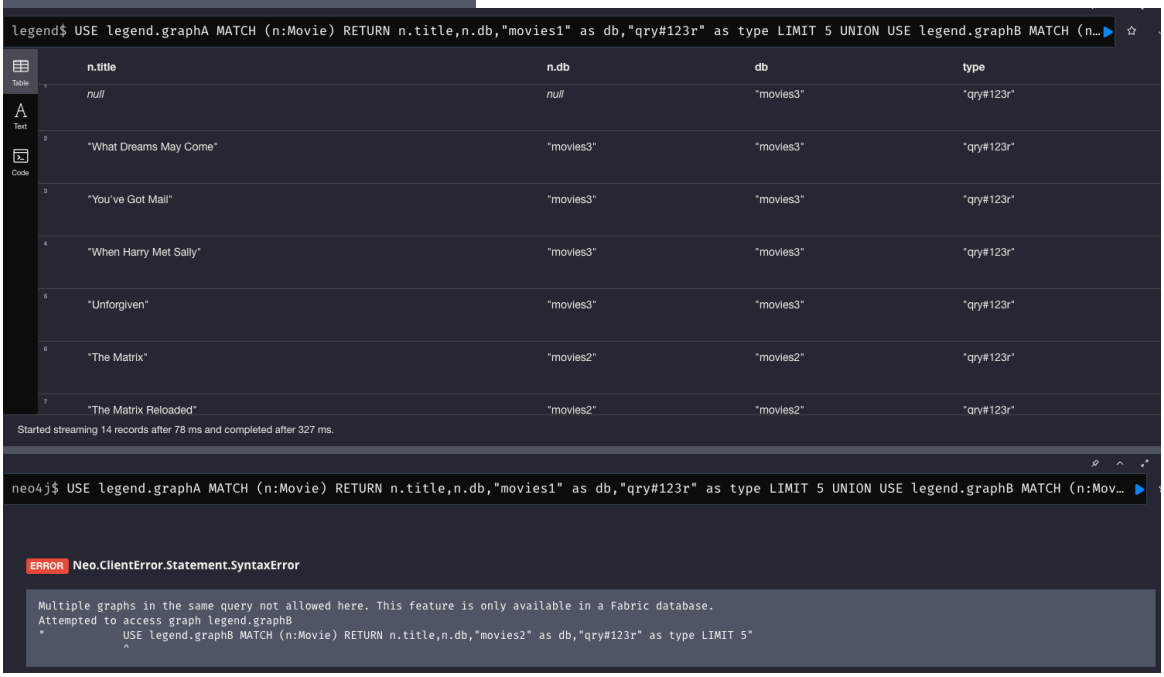
.

1)

4. Running queries without mentioning the database parameter in the session.
   a. {message: Multiple graphs in the same query not allowed here. This feature is only available in a Fabric database.
   Attempted to access graph legend.graphB
   "USE legend.graphB MATCH (n:Movie) RETURN n.title,n.db,"movies2" as db,"qry#123r" as type LIMIT 5"
   ^}
   b. Will discuss query by query here.
   c. On fabric server browser or using ingest: Single data base queries will run by Default "neo4j" or "legend".
   d.

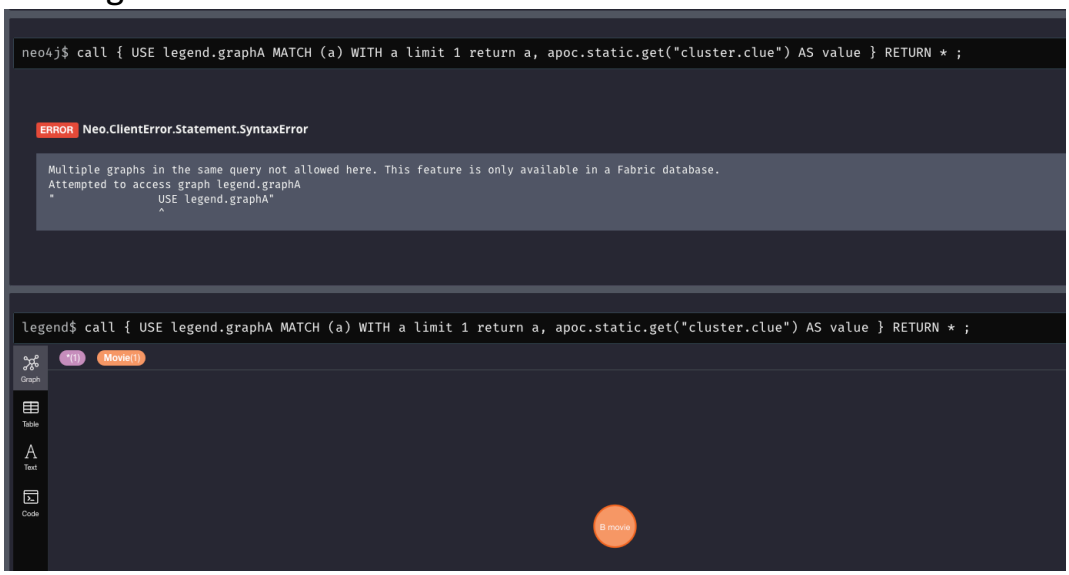


e. On fabric server browser or using ingest: Multi data base queries will run by only fabric databse "legend". If run against "neo4j" will get the error "Multiple graphs in the same query not allowed here. This feature is only available in a Fabric database."



f. About queries using CALL: Queries using CALL should only run against database "legend" which is defined fabric database.

i.



5. Plugins:
   a. If we are using the fabric.routing.servers concept, then we need to install all related plugins in each of the fabric servers.
6. Sharding Implementation challenges:
   a. Sharding criteria could be stored as graph or a function/procedure on the Fabric server. If we are using fabric.routing.servers, these need to be in sync on all servers.
      i. Example queries:

1) Mapping function defined on both the Fabric routing servers:

```
CALL apoc.custom.asFunction(
 'get_shard_db',
 'WITH $movie_name AS movie_name
WITH left(movie_name,1) as title_starting
RETURN CASE WHEN toLower(title_starting)<="n"
THEN 0
 WHEN "o"<=toLower(title_starting)<="t" THEN 1
 ELSE 3 END AS db_name;
',
 'string',
 [['movie_name','string']]
);
```

2) The shading is on the first alphabet of the title. So included that logic to identify the which db it has to go.
3) Retrieval query from corresponding db.

```
WITH "The Da Vinci Code" as title
WITH title,custom.get_shard_db(title) as db_code
CALL {
WITH title,db_code
USE legend.graph(db_code)
MATCH (m:Movie) WHERE m.title = title
RETURN m
}
RETURN *
```

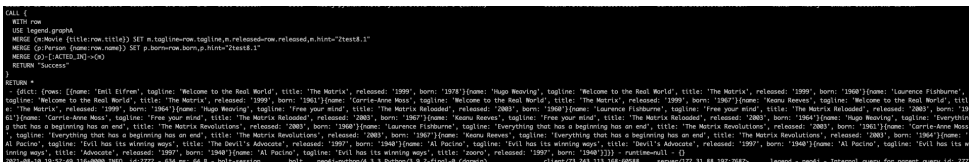
b. Routing context: Each routing context has to be mentioned as a separate database.
   i. Ref: "cluster routing context" in https://neo4j.com/docs/operations-manual/current/fabric/configuration/

7. Supplying the destination data base name for the Session parameter.
   a. database: "legend.graphA" parameter thru config for neo_driver.session(database=database,default_access_mode=default_access_mode)
   b. Result: "Database connection error {code: Neo.ClientError.Database.DatabaseNotFound} {message: Unable to get a routing table for database 'legend.graphA' because this database does not exist}"

c.

```
Database connection error {code: Neo.ClientError.Database.DatabaseNotFound} {message: Unable to get a routing table for database 'legend.graphA' because this database does not exist}
Traceback (most recent call last):
```

8. Loading queries with pyingest as batch
   a. Loading query with CALL:
      i. Fabric server: Submitted as a batch.
         1)



      ii. On CORE server:
         1)

DETACH DELETE `m` - {} - runtime=pipelined - {type: 'user-direct', app: 'neo4j-browser_v4.3.1'}
2021-08-10 19:52:48.679+0000 INFO Query started: id:195 - 56 ms: 0 B - bolt-session bolt neo4j-java/4.3.2 client/3.86.237.103:38032 server/172.31.82.143:7687> movies1 - neo4j - WITH $`@@row` AS `row`
WITH `row` AS `row`
MERGE (`m`:`Movie` {`title`: (`row`).`title`})
SET (`m`).`tagline` = (`row`).`tagline`, (`m`).`released` = (`row`).`released`, (`m`).`hint` = "2test8.1"
MERGE (`p`:`Person` {`name`: (`row`).`name`})
SET (`p`).`born` = (`row`).`born`, (`p`).`hint` = "2test8.1"
MERGE (`p`)-[:`ACTED_IN`]->(`m`)
RETURN "Success" AS `"Success"` - {dict: {rows: [{name: 'Emil Eifrem', tagline: 'Welcome to the Real World', title: 'The Matrix', released: '1999', born: '1978'}{name: 'Hugo Weaving', tagline: 'Welcome to the Real World', title: 'The Matrix', released: '1999', born: '1960'}{name: 'Laurence Fishburne', tagline: 'Welcome to the Real World', title: 'The Matrix', released: '1999', born: '1961'}{name: 'Carrie-Anne Moss', tagline: 'Welcome to the Real World', title: 'The Matrix', released: '1999', born: '1967'}{name: 'Keanu Reeves', tagline: 'Welcome to the Real World', title: 'The Matrix', released: '1999', born: '1964'}{name: 'Hugo Weaving', tagline: 'Free your mind', title: 'The Matrix Reloaded', released: '2003', born: '1960'}{name: 'Laurence Fishburne', tagline: 'Free your mind', title: 'The Matrix Reloaded', released: '2003', born: '1961'}{name: 'Carrie-Anne Moss', tagline: 'Free your mind', title: 'The Matrix Reloaded', released: '2003', born: '1967'}{name: 'Keanu Reeves', tagline: 'Free your mind', title: 'The Matrix Reloaded', released: '2003', born: '1964'}{name: 'Hugo Weaving', tagline: 'Everything that has a beginning has an end', title: 'The Matrix Revolutions', released: '2003', born: '1960'}{name: 'Laurence Fishburne', tagline: 'Everything that has a beginning has an end', title: 'The Matrix Revolutions', released: '2003', born: '1961'}{name: 'Carrie-Anne Moss', tagline: 'Everything that has a beginning has an end', title: 'The Matrix Revolutions', released: '2003', born: '1967'}{name: 'Keanu Reeves', tagline: 'Everything that has a beginning has an end', title: 'The Matrix Revolutions', released: '2003', born: '1964'}{name: 'Al Pacino', tagline: 'Evil has its winning ways', title: 'The Devil's Advocate', released: '1997', born: '1940'}{name: 'Al Pacino', tagline: 'Evil has its winning ways', title: 'Devil's Advocate', released: '1997', born: '1940'}{name: 'Al Pacino', tagline: 'Evil has its winning ways', title: 'Devil's Advocate', released: '1997', born: '1940'}{name: 'Al Pacino', tagline: 'Evil has its winning ways', title: 'zooro', released: '1997', born: '1940'}]}, @@row: {name: 'Hugo Weaving', tagline: 'Welcome to the Real World', title: 'The Matrix', released: '1999', born: '1960'}} - runtime=null - {}
2021-08-10 19:52:48.933+0000 INFO Query started: id:197 - 0 ms: 0 B - bolt-session bolt neo4j-java/4.3.2 client/3.86.237.103:38032 server/172.31.82.143:7687> movies1 - neo4j - WITH $`@@row` AS `row`
WITH `row` AS `row`
MERGE (`m`:`Movie` {`title`: (`row`).`title`})
SET (`m`).`tagline` = (`row`).`tagline`, (`m`).`released` = (`row`).`released`, (`m`).`hint` = "2test8.1"
MERGE (`p`:`Person` {`name`: (`row`).`name`})
SET (`p`).`born` = (`row`).`born`, (`p`).`hint` = "2test8.1"
MERGE (`p`)-[:`ACTED_IN`]->(`m`)
RETURN "Success" AS `"Success"` - {dict: {rows: [{name: 'Emil Eifrem', tagline: 'Welcome to the Real World', title: 'The Matrix', released: '1999', born: '1978'}{name: 'Hugo Weaving', tagline: 'Welcome to the Real World', title: 'The Matrix', released: '1999', born: '1960'}{name: 'Laurence Fishburne', tagline: 'Welcome to the Real World', title: 'The Matrix', released: '1999', born: '1961'}{name: 'Carrie-Anne Moss', tagline: 'Welcome to the Real World', title: 'The Matrix', released: '1999', born: '1967'}{name: 'Keanu Reeves', tagline: 'Welcome to the Real World', title: 'The Matrix', released: '1999', born: '1964'}{name: 'Hugo Weaving', tagline: 'Free your mind', title: 'The Matrix Reloaded', released: '2003', born: '1960'}{name: 'Laurence Fishburne', tagline: 'Free your mind', title: 'The Matrix Reloaded', released: '2003', born: '1961'}{name: 'Carrie-Anne Moss', tagline: 'Free your mind', title: 'The Matrix Reloaded', released: '2003', born: '1967'}{name: 'Keanu Reeves', tagline: 'Free your mind', title: 'The Matrix Reloaded', released: '2003', born: '1964'}{name: 'Hugo Weaving', tagline: 'Everything that has a beginning has an end', title: 'The Matrix Revolutions', released: '2003', born: '1960'}{name: 'Laurence Fishburne', tagline: 'Everything that has a beginning has an end', title: 'The Matrix Revolutions', released: '2003', born: '1961'}{name: 'Carrie-Anne Moss', tagline: 'Everything that has a beginning has an end', title: 'The Matrix Revolutions', released: '2003', born: '1967'}{name: 'Keanu Reeves', tagline: 'Everything that has a beginning has an end', title: 'The Matrix Revolutions', released: '2003', born: '1964'}{name: 'Al Pacino', tagline: 'Evil has its winning ways', title: 'The Devil's Advocate', released: '1997', born: '1940'}{name: 'Al Pacino', tagline: 'Evil has its winning ways', title: 'Devil's Advocate', released: '1997', born: '1940'}{name: 'Al Pacino', tagline: 'Evil has its winning ways', title: 'zooro', released: '1997', born: '1940'}]} - runtime=null - {}

```
CALL {
  WITH row
  USE legend.graphA
  MERGE (m:Movie {title:row.title}) SET m.tagline=row.tagline,m.released=row.released,m.hint="2test8.1"
  MERGE (p:Person {name:row.name}) SET p.born=row.born,p.hint="2test8.1"
  MERGE (p)-[:ACTED_IN]->(m)
  RETURN "Success"
}
RETURN *
```

- {dict: {rows: [{name: 'Emil Eifrem', tagline: 'Welcome to the Real World', title: 'The Matrix', released: '1999', born: '1978'}{name: 'Hugo Weaving', tagline: 'Welcome to the Real World', title: 'The Matrix', released: '1999', born: '1960'}{name: 'Laurence Fishburne', tagline: 'Welcome to the Real World', title: 'The Matrix', released: '1999', born: '1961'}{name: 'Carrie-Anne Moss', tagline: 'Welcome to the Real World', title: 'The Matrix', released: '1999', born: '1967'}{name: 'Keanu Reeves', tagline: 'Welcome to the Real World', title: 'The Matrix', released: '1999', born: '1964'}{name: 'Hugo Weaving', tagline: 'Free your mind', title: 'The Matrix Reloaded', released: '2003', born: '1960'}{name: 'Laurence Fishburne', tagline: 'Free your mind', title: 'The Matrix Reloaded', released: '2003', born: '1961'}{name: 'Carrie-Anne Moss', tagline: 'Free your mind', title: 'The Matrix Reloaded', released: '2003', born: '1967'}{name: 'Keanu Reeves', tagline: 'Free your mind', title: 'The Matrix Reloaded', released: '2003', born: '1964'}{name: 'Hugo Weaving', tagline: 'Everything that has a beginning has an end', title: 'The Matrix Revolutions', released: '2003', born: '1960'}{name: 'Laurence Fishburne', tagline: 'Everything that has a beginning has an end', title: 'The Matrix Revolutions', released: '2003', born: '1961'}{name: 'Carrie-Anne Moss', tagline: 'Everything that has a beginning has an end', title: 'The Matrix Revolutions', released: '2003', born: '1967'}{name: 'Keanu Reeves', tagline: 'Everything that has a beginning has an end', title: 'The Matrix Revolutions', released: '2003', born: '1964'}{name: 'Al Pacino', tagline: 'Evil has its winning ways', title: 'The Devil's Advocate', released: '1997', born: '1940'}{name: 'Al Pacino', tagline: 'Evil has its winning ways', title: 'Devil's Advocate', released: '1997', born: '1940'}{name: 'Al Pacino', tagline: 'Evil has its winning ways', title: 'zooro', released: '1997', born: '1940'}]} - runtime=null - {}

Matrix Reloaded', released: '2003', born: '1967'}{name: 'Keanu Reeves', tagline: 'Free your mind', title: 'The Matrix Reloaded', released: '2003', born: '1964'}{name: 'Hugo Weaving', tagline: 'Everything that has a beginning has an end', title: 'The Matrix Revolutions', released: '2003', born: '1960'}{name: 'Laurence Fishburne', tagline: 'Everything that has a beginning has an end', title: 'The Matrix Revolutions', released: '2003', born: '1961'} {name: 'Carrie-Anne Moss', tagline: 'Everything that has a beginning has an end', title: 'The Matrix Revolutions', released: '2003', born: '1967'}{name: 'Keanu Reeves', tagline: 'Everything that has a beginning has an end', title: 'The Matrix Revolutions', released: '2003', born: '1964'}{name: 'Al Pacino', tagline: 'Evil has its winning ways', title: 'The Devil's Advocate', released: '1997', born: '1940'}{name: 'Al Pacino', tagline: 'Evil has its winning ways', title: 'Devil's Advocate', released: '1997', born: '1940'}{name: 'Al Pacino', tagline: 'Evil has its winning ways', title: 'Advocate', released: '1997', born: '1940'}{name: 'Al Pacino', tagline: 'Evil has its winning ways', title: 'zooro', released: '1997', born: '1940'}]}, @@row: {name: 'Emil Eifrem', tagline: 'Welcome to the Real World', title: 'The Matrix', released: '1999', born: '1978'}} - runtime=null - {}

iii. Observation: query has been submitted for each row as expected. But sending total data every time and the updating row as "@@row"

b. Loading query without CALL:

i. Result: As expected total data has been passed to Core server for execution. But each query was logged twice in query log on fabric server as well as core server.

Fabric server:

2021-08-10 20:24:51.070+0000 INFO Query started: id:2780 - 17 ms: 0 B - bolt-session bolt neo4j-python/4.3.3 Python/3.9.2-final-0 (darwin) client/73.243.113.168:60837 server/172.31.88.197:7687> <none> - neo4j - USE legend.graphB
UNWIND $dict.rows as row
MERGE (m:Movie {title:row.title}) SET m.tagline=row.tagline,m.released=row.released,m.hint="3test8.4"
MERGE (p:Person {name:row.name}) SET p.born=row.born,p.hint="3test8.4"
MERGE (p)-[:ACTED_IN]->(m)
RETURN "Success"
- {dict: {rows: [{name: 'Emil Eifrem', tagline: 'Welcome to the Real

- {dict: {rows: [{name: 'Emil Eifrem', tagline: 'Welcome to the Real World', title: 'The Matrix', released: '1999', born: '1978'}{name: 'Hugo Weaving', tagline: 'Welcome to the Real World', title: 'The Matrix', released: '1999', born: '1960'}{name: 'Laurence Fishburne', tagline: 'Welcome to the Real World', title: 'The Matrix', released: '1999', born: '1961'}{name: 'Carrie-Anne Moss', tagline: 'Welcome to the Real World', title: 'The Matrix', released: '1999', born: '1967'}{name: 'Keanu Reeves', tagline: 'Welcome to the Real World', title: 'The Matrix', released: '1999', born: '1964'} {name: 'Hugo Weaving', tagline: 'Free your mind', title: 'The Matrix Reloaded', released: '2003', born: '1960'}{name: 'Laurence Fishburne', tagline: 'Free your mind', title: 'The Matrix Reloaded', released: '2003', born: '1961'}{name: 'Carrie-Anne Moss', tagline: 'Free your mind', title: 'The Matrix Reloaded', released: '2003', born: '1967'}{name: 'Keanu Reeves', tagline: 'Free your mind', title: 'The Matrix Reloaded', released: '2003', born: '1964'}{name: 'Hugo Weaving', tagline: 'Everything that has a beginning has an end', title: 'The Matrix Revolutions', released: '2003', born: '1960'} {name: 'Laurence Fishburne', tagline: 'Everything that has a beginning has an end', title: 'The Matrix Revolutions', released: '2003', born: '1961'}{name: 'Carrie-Anne Moss', tagline: 'Everything that has a beginning has an end', title: 'The Matrix Revolutions', released: '2003', born: '1967'}{name: 'Keanu Reeves', tagline: 'Everything that has a beginning has an end', title: 'The Matrix Revolutions', released: '2003', born: '1964'}{name: 'Al Pacino', tagline: 'Evil has its winning ways', title: 'The Devil's Advocate', released: '1997', born: '1940'}{name: 'Al Pacino', tagline: 'Evil has its winning ways', title: 'Devil's Advocate', released: '1997', born: '1940'}{name: 'Al Pacino', tagline: 'Evil has its winning ways', title: 'Advocate', released: '1997', born: '1940'} {name: 'Al Pacino', tagline: 'Evil has its winning ways', title: 'zooro', released: '1997', born: '1940'}]}} - runtime=null - {}
2021-08-10 20:24:51.784+0000 INFO id:2780 - 731 ms: -1 B - bolt-session bolt neo4j-python/4.3.3 Python/3.9.2-final-0 (darwin) client/73.243.113.168:60837 server/172.31.88.197:7687> <none> - neo4j - USE legend.graphB
UNWIND $dict.rows as row
MERGE (m:Movie {title:row.title}) SET m.tagline=row.tagline,m.released=row.released,m.hint="3test8.4"
MERGE (p:Person {name:row.name}) SET p.born=row.born,p.hint="3test8.4"

p.born=row.born,p.hint="$test8.4"
MERGE (p)-[:ACTED_IN]->(m)
RETURN "Success"
 - {dict: {rows: [{name: 'Emil Eifrem', tagline: 'Welcome to the Real World', title: 'The Matrix', released: '1999', born: '1978'}{name: 'Hugo Weaving', tagline: 'Welcome to the Real World', title: 'The Matrix', released: '1999', born: '1960'}{name: 'Laurence Fishburne', tagline: 'Welcome to the Real World', title: 'The Matrix', released: '1999', born: '1961'}{name: 'Carrie-Anne Moss', tagline: 'Welcome to the Real World', title: 'The Matrix', released: '1999', born: '1967'}{name: 'Keanu Reeves', tagline: 'Welcome to the Real World', title: 'The Matrix', released: '1999', born: '1964'} {name: 'Hugo Weaving', tagline: 'Free your mind', title: 'The Matrix Reloaded', released: '2003', born: '1960'}{name: 'Laurence Fishburne', tagline: 'Free your mind', title: 'The Matrix Reloaded', released: '2003', born: '1961'}{name: 'Carrie-Anne Moss', tagline: 'Free your mind', title: 'The Matrix Reloaded', released: '2003', born: '1967'}{name: 'Keanu Reeves', tagline: 'Free your mind', title: 'The Matrix Reloaded', released: '2003', born: '1964'}{name: 'Hugo Weaving', tagline: 'Everything that has a beginning has an end', title: 'The Matrix Revolutions', released: '2003', born: '1960'} {name: 'Laurence Fishburne', tagline: 'Everything that has a beginning has an end', title: 'The Matrix Revolutions', released: '2003', born: '1961'}{name: 'Carrie-Anne Moss', tagline: 'Everything that has a beginning has an end', title: 'The Matrix Revolutions', released: '2003', born: '1967'}{name: 'Keanu Reeves', tagline: 'Everything that has a beginning has an end', title: 'The Matrix Revolutions', released: '2003', born: '1964'}{name: 'Al Pacino', tagline: 'Evil has its winning ways', title: 'The Devil's Advocate', released: '1997', born: '1940'}{name: 'Al Pacino', tagline: 'Evil has its winning ways', title: 'Devil's Advocate', released: '1997', born: '1940'}{name: 'Al Pacino', tagline: 'Evil has its winning ways', title: 'Advocate', released: '1997', born: '1940'} {name: 'Al Pacino', tagline: 'Evil has its winning ways', title: 'zooro', released: '1997', born: '1940'}]}} - runtime=null - {}

Core server: 8
2021-08-10 20:24:51.216+0000 INFO Query started: id:9 - 118 ms: 0 B - bolt-session	bolt	neo4j-java/4.3.2	client/3.86.237.103:38052	server/172.31.82.143:7687>	movies2 - neo4j - UNWIND ($`dict`).`rows` AS `row`

MERGE (`m`:`Movie` {`title`: (`row`).`title`})
SET (`m`).`tagline` = (`row`).`tagline`, (`m`).`released` = (`row`).`released`, (`m`).`hint` = "3test8.4"
MERGE (`p`:`Person` {`name`: (`row`).`name`})
SET (`p`).`born` = (`row`).`born`, (`p`).`hint` = "3test8.4"
MERGE (`p`)-[:`ACTED_IN`]->(`m`)
RETURN "Success" AS `"Success"` - {dict: {rows: [{name: 'Emil Eifrem', tagline: 'Welcome to the Real World', title: 'The Matrix', released: '1999', born: '1978'}{name: 'Hugo Weaving', tagline: 'Welcome to the Real World', title: 'The Matrix', released: '1999', born: '1960'}{name: 'Laurence Fishburne', tagline: 'Welcome to the Real World', title: 'The Matrix', released: '1999', born: '1961'}{name: 'Carrie-Anne Moss', tagline: 'Welcome to the Real World', title: 'The Matrix', released: '1999', born: '1967'}{name: 'Keanu Reeves', tagline: 'Welcome to the Real World', title: 'The Matrix', released: '1999', born: '1964'}{name: 'Hugo Weaving', tagline: 'Free your mind', title: 'The Matrix Reloaded', released: '2003', born: '1960'}{name: 'Laurence Fishburne', tagline: 'Free your mind', title: 'The Matrix Reloaded', released: '2003', born: '1961'}{name: 'Carrie-Anne Moss', tagline: 'Free your mind', title: 'The Matrix Reloaded', released: '2003', born: '1967'}{name: 'Keanu Reeves', tagline: 'Free your mind', title: 'The Matrix Reloaded', released: '2003', born: '1964'}{name: 'Hugo Weaving', tagline: 'Everything that has a beginning has an end', title: 'The Matrix Revolutions', released: '2003', born: '1960'}{name: 'Laurence Fishburne', tagline: 'Everything that has a beginning has an end', title: 'The Matrix Revolutions', released: '2003', born: '1961'}{name: 'Carrie-Anne Moss', tagline: 'Everything that has a beginning has an end', title: 'The Matrix Revolutions', released: '2003', born: '1967'}{name: 'Keanu Reeves', tagline: 'Everything that has a beginning has an end', title: 'The Matrix Revolutions', released: '2003', born: '1964'}{name: 'Al Pacino', tagline: 'Evil has its winning ways', title: 'The Devil's Advocate', released: '1997', born: '1940'}{name: 'Al Pacino', tagline: 'Evil has its winning ways', title: 'Devil's Advocate', released: '1997', born: '1940'}{name: 'Al Pacino', tagline: 'Evil has its winning ways', title: 'Advocate', released: '1997', born: '1940'}{name: 'Al Pacino', tagline: 'Evil has its winning ways', title: 'zooro', released: '1997', born: '1940'}]}} - runtime=null - {}
2021-08-10 20:24:51.777+0000 INFO id:9 - 679 ms: 15444 B - bolt-session bolt neo4j-java/4.3.2

client/3.86.237.103:38052 server/172.31.82.143:7687>
movies2 - neo4j - UNWIND ($`dict`).`rows` AS `row`
MERGE (`m`:`Movie` {`title`: (`row`).`title`})
SET (`m`).`tagline` = (`row`).`tagline`, (`m`).`released` = (`row`).
`released`, (`m`).`hint` = "3test8.4"
MERGE (`p`:`Person` {`name`: (`row`).`name`})
SET (`p`).`born` = (`row`).`born`, (`p`).`hint` = "3test8.4"
MERGE (`p`)-[:`ACTED_IN`]->(`m`)
RETURN "Success" AS `"Success"` - {dict: {rows: [{name: 'Emil Eifrem', tagline: 'Welcome to the Real World', title: 'The Matrix', released: '1999', born: '1978'}{name: 'Hugo Weaving', tagline: 'Welcome to the Real World', title: 'The Matrix', released: '1999', born: '1960'}{name: 'Laurence Fishburne', tagline: 'Welcome to the Real World', title: 'The Matrix', released: '1999', born: '1961'} {name: 'Carrie-Anne Moss', tagline: 'Welcome to the Real World', title: 'The Matrix', released: '1999', born: '1967'}{name: 'Keanu Reeves', tagline: 'Welcome to the Real World', title: 'The Matrix', released: '1999', born: '1964'}{name: 'Hugo Weaving', tagline: 'Free your mind', title: 'The Matrix Reloaded', released: '2003', born: '1960'}{name: 'Laurence Fishburne', tagline: 'Free your mind', title: 'The Matrix Reloaded', released: '2003', born: '1961'} {name: 'Carrie-Anne Moss', tagline: 'Free your mind', title: 'The Matrix Reloaded', released: '2003', born: '1967'}{name: 'Keanu Reeves', tagline: 'Free your mind', title: 'The Matrix Reloaded', released: '2003', born: '1964'}{name: 'Hugo Weaving', tagline: 'Everything that has a beginning has an end', title: 'The Matrix Revolutions', released: '2003', born: '1960'}{name: 'Laurence Fishburne', tagline: 'Everything that has a beginning has an end', title: 'The Matrix Revolutions', released: '2003', born: '1961'} {name: 'Carrie-Anne Moss', tagline: 'Everything that has a beginning has an end', title: 'The Matrix Revolutions', released: '2003', born: '1967'}{name: 'Keanu Reeves', tagline: 'Everything that has a beginning has an end', title: 'The Matrix Revolutions', released: '2003', born: '1964'}{name: 'Al Pacino', tagline: 'Evil has its winning ways', title: 'The Devil's Advocate', released: '1997', born: '1940'}{name: 'Al Pacino', tagline: 'Evil has its winning ways', title: 'Devil's Advocate', released: '1997', born: '1940'}{name: 'Al Pacino', tagline: 'Evil has its winning ways', title: 'Advocate', released: '1997', born: '1940'}{name: 'Al Pacino', tagline: 'Evil has its winning ways', title: 'zooro', released: '1997', born: '1940'}]}} - runtime=pipelined - {}

9. Sharding queries:
    a. Query with data going to same database:
        i. When Data was sending to the same shard by shard identifying function, query was running fine.

    b. Query with data going to Different Database:
        i. Error:
        neo4j.exceptions.ClientError: {code: Neo.ClientError.Statement.AccessMode} {message: Writing to more than one database per transaction is not allowed. Attempted write to External{graphId=0, uuid= 00000000-0000-0000-0000-000000000000, databaseName='movies1', uri=RemoteUri{scheme='neo4j', addresses=[3.84.141.38:7687, 54.210.27.49:7687, 52.87.176.179:7687], query='null'}}, currently writing to External{graphId=1, uuid= 00000000-0000-0001-0000-000000000000, databaseName='movies2', uri=RemoteUri{scheme='neo4j', addresses=[3.84.141.38:7687, 54.210.27.49:7687, 52.87.176.179:7687], query='null'}}}
    c. Observation: when the data is for only one shard, query worked fine. When data modified to go for different shards, though query is sending row by row, this error came.

Try next:

1. Unwind without call, unwind with call using pyingest - done
2. Send the exact database in the session and try to run queries without mentioning USE. - Done
3. Sharding with writes. - Done
4. Configure Bloom. Need to keep sw in all the servers. --> check it in team bloom