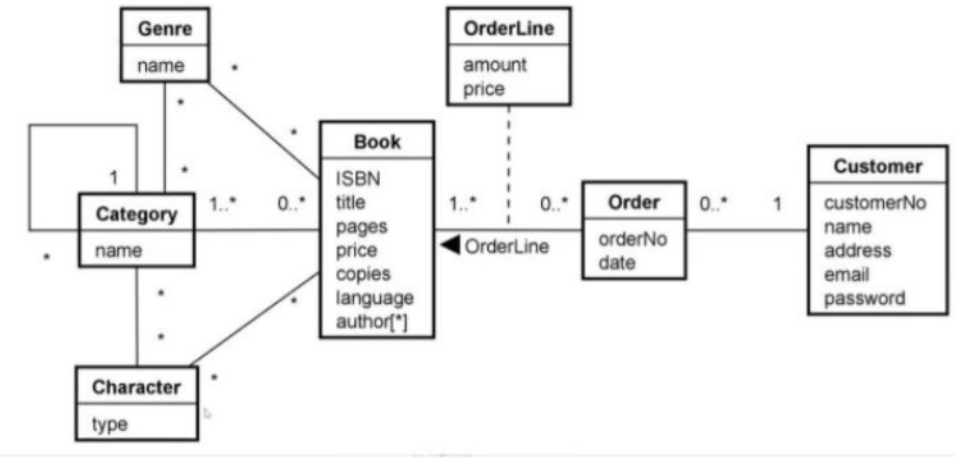
NSQ1 – Assignment 3

Students:
Florin – Leonard Bordei, 280593
Jaume Lopez Topping, 282231.

Question 01 – Model Database



Question 02 – Create schemas

Create books

CREATE (:Book{ISBN: 10000001,title: "Interestellar",pages: 10, price: 20,copies:11, language: "English"})

CREATE (:Book{ISBN: 10000002,title: "Jungle Book",pages: 333, price: 100,copies:12, language: "English"})

CREATE (:Book{ISBN: 10000003,title: "Apollo 12",pages: 335, price: 50,copies:20, language: "English"})

CREATE (:Book{ISBN: 10000004,title: "Final Destiny",pages: 350, price: 55,copies:99, language: "English"})

```
CREATE (:Book{ISBN: 10000005,title: "Harry Potter",pages: 444, price: 55,copies:220, language: "English"})
```

Categories

```
CREATE (Fiction:Category{ categoryName:"Fiction"})
    CREATE (ScienceFiction:Category{ categoryName:"Science Fiction"})
        CREATE (SpaceOpera:Categoary{ categoryName:"Space Opera"})
        CREATE (CyberPunk:Category{ categoryName:"Cyber Punk"})
        CREATE (Dystopian:Category{ categoryName:"Dystopian"})


CREATE (ScienceFictionFantasy:Category{ categoryName:"Science Fiction & Fantasy"})

CREATE (NonFiction:Category{ categoryName:"Non Fiction"})

CREATE (Fantasy:Category{ categoryName:"Fantasy"})
```

```
MATCH (a:Category),(b:Category)
WHERE a.categoryName = "Science Fiction & Fantasy" AND b.categoryName = "Science Fiction"
CREATE (a)-[r:Children]->(b)
RETURN r

MATCH (a:Category),(b:Category)
WHERE a.categoryName = "Science Fiction" AND b.categoryName = "Space Opera"
CREATE (a)-[r:Children]->(b)
RETURN r
```

```
MATCH (a:Category),(b:Category)
WHERE a.categoryName = "Science Fiction" AND b.categoryName = "Cyber Punk"
CREATE (a)-[r:Children]->(b)
RETURN r

MATCH (a:Category),(b:Category)
WHERE a.categoryName = "Science Fiction" AND b.categoryName = "Dystopian"
CREATE (a)-[r:Children]->(b)
RETURN r

MATCH (a:Category),(b:Category)
WHERE a.categoryName = "Fantasy" AND b.categoryName = "Science Fiction & Fantasy"
CREATE (a)-[r:Children]->(b)
RETURN r

MATCH (a:Category),(b:Category)
WHERE a.categoryName = "Fiction" AND b.categoryName = "Science Fiction"
CREATE (a)-[r:Children]->(b)
RETURN r

MATCH (a:Category),(b:Category)
WHERE a.categoryName = "Science Fiction & Fantasy" AND b.categoryName = "Science
Fiction"
CREATE (a)-[r:Children]->(b)
RETURN r
```

Question 03: Work with data

Modifying data

1. Sell a book to a customer

```
CREATE (o:Order{orderNo:1,date:"2020-12-20"})-[:CONTAINS]-
>(ol:Orderline{lineNo:01,amount:2,price:110})

MATCH (ol:Orderline {lineNo:01}), (b:Book {ISBN:10000005})

CREATE (ol)-[:INCLUDES]->(b)
```

2. Change the address of a customer

   match (c:Customer{cust_no:17}) set c.address = "Changed address no 20" return c.address

3. Add an existing author to a book
   match (a:Author{auth_name:"Emily Dickinson" }), (b:Book{title:"Interestellar"})
   CREATE (a) -[r:WROTE] ->(b) return r

4. Retire the space opera category and assign all books from that category to the parent category, don't assume you know the id of the parent category

```
MATCH (b:Book {ISBN:10000001}), (c:Category {categoryName:"Space Opera"})

CREATE (b)-[:IS]->(c)
```

```
MATCH  (b:Book)-[:IS]->(c:Category {categoryName:"Space Opera"}), (d:Category)-
[Children]->(:Category {categoryName:"Space Opera"})
CREATE (b)-[:IS]->(d)
```

```
MATCH (c:Category{categoryName:"Space Opera"}) DETACH DELETE c
```

5. Change a book from not fiction to fiction

```
    MATCH (b:Book {ISBN:10000003}), (c:Category {categoryName:"Non Fiction"}
)
    CREATE (b)-[:IS]->(c)
```

```
    MATCH (b:Book {ISBN:10000003}), (c:Category {categoryName:"Fiction"})
```

```
CREATE (b)-[:IS]->(c)
```

```
MATCH (b:Book {ISBN:10000003})-[r:IS]-
>(c:Category {categoryName:"Non Fiction"}) delete r
```

6. Retire the Fantasy & SF category and just use either fantasy or sci fy

7. Sell 3 copies of 1 book and 2 of another in one single order

   Create (:Order{orderNo:2, date:"2020-11-20"})
   create (:Orderline{lineNo:02, amount:3, price:300})

   MATCH (o:Order{orderNo:2}), (ol1:Orderline{lineNo:01}), (ol2:Orderline{lineNo:02})
   CREATE (ol1)<-[:CONTAINS]-(o)-[:CONTAINS]->(ol2) return o

   match (b:Book{title:"Jungle Book"}), (ol:Orderline {lineNo:02}) set b.copies = b.copies - ol.amount return b.copies
   match (b:Book{title:"Harry Potter"}), (ol:Orderline {lineNo:01}) set b.copies = b.copies - ol.amount return b.copies

Querying data

1. All books by an author
   Match (:Author{auth_name:"Emily Dickinson"})-[:WROTE]->(b:Book) return b.title

2. Total price of an order
   MATCH (ol:Orderline)<-[:CONTAINS]-(o:Order{orderNo:2}) RETURN sum(ol.price) AS totalPrice

3. Total sales to a customer
   Match (c:Customer{cust_name:"Nick Diaz"}), (o:Order{orderNo:2}) CREATE (c)-[:BOUGHT]->(o)
   Match (ol:Orderline)<-[:CONTAINS]-(o:Order)<-[:BOUGHT]-(:Customer{cust_name:"Nick Diaz"}) return sum(ol.price) as totalPrice

4. Books that are categorized as neither fiction or non fiction
   MATCH (b:Book),(c:Category) WHERE (b)-[:IS]->(c) and c.categoryName <> "Fiction" and c.categoryName<>"Non Fiction" return b

5. Average page count by genre
   match (b:Book{title:"Harry Potter"}), (g:Genre{name:"adventure"}) CREATE (b)-[:HAS_GENRE] -> (g)
   match (b:Book{title:"Jungle Book"}), (g:Genre{name:"adventure"}) CREATE (b)-[:HAS_GENRE] -> (g)
   MATCH (b:Book)-[:HAS_GENRE]->(g:Genre{name:"adventure"}) RETURN avg(b.pages) as avgPages

6. Categories that have no sub categories
   MATCH (c:Category) WHERE NOT (c)-[:Children]->() return c

7. ISBN numbers of books with more than one author
   MATCH (a:Author)-[:WROTE]->(b:Book) WITH collect(b.ISBN) as books WHERE size(books)>1 return books

8. ISBN numbers of books that sold at least x copies
   MATCH (b:Book)<-[:INCLUDES]-(o:Orderline) WHERE o.amount>1 return b.ISBN

9. Best-selling books: The top 10 selling books ordered in descending order by number of sales.
   MATCH (b:Book),(ol:Orderline) where (ol)-[:INCLUDES]->(b) return sum(ol.amount) as qtysold,b ORDER BY qtysold DESC
   LIMIT 10;

10. Best-selling genres: The top 3 selling genres ordered in descending order by number of sales.

MATCH (ol:Orderline)-[:INCLUDES]->(b:Book)-[:IS]->(c:Category) return sum(ol.amount),c.categoryName as totalAmount ORDER BY totalAmount LIMIT 3

11. All science fiction books (Hint: $graphLookup)

    MATCH (b:Book)-[:IS]->(c:Category {categoryName: "Science Fiction"}) return b.title

12. Characters used in science fiction books

    MATCH (c:Character)-[:ACTED]->(b:Book)-[:IS]->(:C
ategory {categoryName: "Science Fiction"}) return c

13. Number of books in each category

    match (b:Book)-[r:IS]->(c:Category) return count(r) as numberOfBooks,c.categoryName

15. Suggestions for other books that the customer might like to buy (This is deliberately open. Find your own criteria.)

Question 04 – Report

Question 1-3
What were the decisions taken in the modelling?
Why were these decisions taken?
What were the consequences of these decisions?

The modelling of the graph database started from translating the relational database into a graph one. Therefore, each table represented a node with certain labels (Book, category etc.). After setting up the nodes we've arranged the relationships between the nodes, each relationship having a name example: Order - contains -> orderline. The foreign or the primary keys have not been included as they are not to be used in a graph database.

Question 4: What were the difficult and easy parts of the exercise?

Some exercises where we had to use the aggregate functions represented some small difficulties while the exercises where we had to change properties in a node have been the least difficult ones.

Question 5: How does that compare to relational database?

The relational database is more of a strict structure of data while the graph one is more flexible and gives more freedom while working with the data.

Question 6: What are the advantages and disadvantages of Neo4J over relational databases and MongoDb for this exercise?

Comparing with both, relational database and MongoDb, it is much easier to associate the relations, the data visualization on queries, the speed of computing and the use of aggregate functions it is easier.
While the disadvantages are that there is a limitation on how many nodes are to be added, sharding is not supported and it is difficult to scale.