# NSQ - 2nd assignment report

*Note: The queries can be tested as we have the database hosted:*

*Connection string:*
*mongodb+srv://nsq1:ypmX67TYXeXuzMc9@cluster0.ktu5a.mongodb.net/2ND_ASSIGNMENT?retryWrites=true&w=majority*

Michal Martincek - 279954
Rafał Pierścieniak - 279471
Dalius Seminauskas - 285039
Adrian Ionut Patrascu - 280188

- *What were the decisions taken in the modelling?*
- *Why were these decisions taken?*
- *What were the consequences of these decisions?*
- *What were the difficult and easy parts of the exercise?*
- *How does that compare to relational databases?*
- *What are the advantages and disadvantages of MongoDB over relational databases for this exercise*

The model was based on the learning from previous assignment which utilised a traditional SQL-based model. We transformed this into a model that is more suitable for the No-SQL approach taken in this assignment. We did that by removing lookup tables and reducing the number of dependencies on other tables.

We decided to use four collections to store the data: Books, Categories, Orders and Customers. Their respective schemas can be seen in the *database_schema.mongo* file attached to this report. Part of the model has a direct reference to the corresponding data such as the list of authors for each book since we only need their names and nothing else.

On the other hand data such as the book's categories have a reference to each category they belong to. This was done to prevent unnecessary duplicate data and to streamline the model. Other fields use this approach as well. For example to keep track of each book and its quantity for each order we store only the id and sold quantity of that book.

The consequence of this approach is that while we keep some duplicate data it's only the bare minimum which on its own would result in unnecessary complexity if separated into its own collection/s. This approach also allows us to change the underlying data which is

stored using the reference method described above. For example when we want to update the subcategories of all sci-fi books we only need to do so in one place.

The document based nature of MongoDB might lead to thinking that it is a good approach to store all data in one single document but while working on this assignment we came to the conclusion that this leads to a cluttered and difficult to work with data that is harder to maintain, update and validate.

NOTE: Populating data into a new database, books will not have a relation to the categories, since category ids are hardcoded to the books, and during the population of the categories, category ids are automatically generated. To solve this, books would have to contain a category name, instead of its id.

**Easy and hard parts of assignment.**

One of the hard parts for us came later in the process of making the assignment when we had to figure out how to set up the database with a sharded cluster considering that it' wasn't very intuitive even though we followed the tutorial. We also had to make a few fixes in order to make some joins for the last queries.

On the other hand, it was easy for us to decide what's more relevant to keep from the previous assignment in order to have a good approach when making a suitable modelling for this assignment.

**MongoDB over relational databases**

Advantages:

- Modelling is easier/faster
- Usually you end up having less tables
- Flexible (you can make not strict schema or none at all)
- Easy setup of shards and replicas - was hard for us since we did it for the first time. In relational databases would be more difficult

Disadvantages:

- If you do not not make strict schema - you might end up having something unexpected in your database as input (extra/missing dimension on tuple)
- Limit of data load per file (16 mb)