**System Design**

**Architecture**

There are several possible database management systems that we can use for this project. We are looking for a managements system that will able to hold a large amount of data which will be accessed by users to view or edit articles. We have come to decide on two systems. MongoDB vs MySQL.

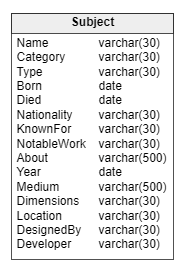
MySQL stores tables in the database which contain attributes within a defined structure represented by a specific datatype. As mysql is a relational database management system, it relies on having tables with relationships to other tables to connect data. MySQL is good for having a structured database with your schema defined.

MongoDB uses documents to store in a collection which do not need to have a defined structure. All related data are stored together in a single document without the need to have any relationships with any other tables or documents. New fields can be added to the document at any time without the need to restructure the entire table. Mongo also has slight performance increases as it doesn't rely on any joins for other data.

We will be using **MongoDB** as our database management system for this project. Its benefits for our database outweigh using MySQL for reasons such as not needing a defined structure. Articles in our database have different structures for each one so this will be easier to manage and design.

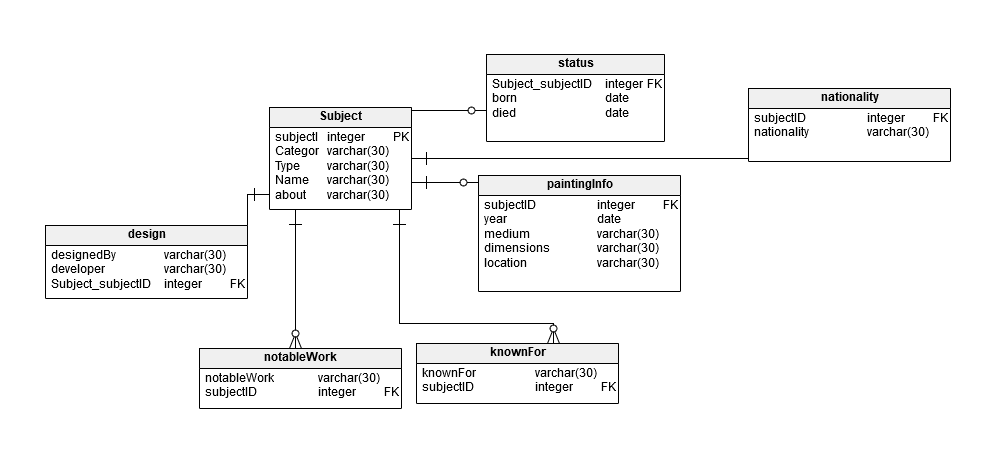
**Database Design**

Since we are using MongoDB having a relational table would be unnecessary. Everything will be under one collection.



Will also require a subject/article id.

If we were using mySQL we would have something like this.



This seems a lot messier for something so simple which is why we will stick to using mongoDB.

**Queries**

**Students**

**Browse article by specific category**

Select \* from article where category = searchCategory

Db.Article.find({category:”searchCategory”}, {\_id:0})

**Browse article by specific keyword in the title? (type, name, any word in article?)**

Select \* from article where about like %searchword%

Db.article.find({about:/searchword/})

**Tutors**

**Add or modify article**

**ADD**

Db.article.insertone({category:”arts”, type:”biography”, name”test”, about”testestetstset”})

**MODIFY**

Db.article.update({subjectID:1}, {$set:{category:”mathematics”, name:”test2”, about:”testststst”})

**Admin**

**Add, modify or remove article**

**REMOVE**

Db.articles.delete({subjectID:1});

**User Access Design**

User security through MongoDB will provide this database with security features such as Authorization, authentication, Encryption and Auditing. We will go over the first two, Authentication and Authorization.

Authentication will be use do identify the user who is accessing the database and prevent access for anyone is can’t be identified. For authentication, we will be using a login system with a unique username and password for every user who will access the database.

Authorization will be used to separate the users into specific groups that can do the defined actions. The groups will be separated where some users will only be able to view the database whereas others will be able to modify the entire database.

There will be three groups which that users will be categorized in.

Student- Should only be able to view the database

Tutor-Should be able to view, add and modify the database

Administrator-Should be able to view,, add, modify and delete items in the database.

**User Experience Design**

The mobile application should have several views such as:

Login page – will authenticate the users and will redirect when the have passed the authentication. This should also have a registration link.

Main page student – will have a search bar where the user will be able to search by category or a keyword in the title. There should also be a list of all the articles by name.

Main page tutor – will have a list of all the articles where they can add or modify the database

Main page admin – will have a list of all the articles where they can add, modify or delete an article in the database.

Article page – will have the article fully displayed with all the information layed out on the page.