MongoDB Installation: [installation](https://www.mongodb.com/docs/manual/tutorial/install-mongodb-on-windows/)

Create an Web API using Mongo: [Creating Web API](https://docs.microsoft.com/en-us/aspnet/core/tutorials/first-mongo-app?view=aspnetcore-6.0&tabs=visual-studio)

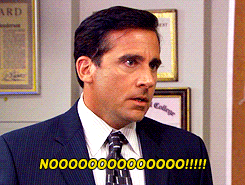
Mongo Schema: [Schema](https://www.mongodb.com/developer/products/mongodb/mongodb-schema-design-best-practices/)

# Schema Design Approaches – Relational vs. MongoDB

When it comes to MongoDB database schema design, this is what most developers think of when they are looking at designing a relational schema and a MongoDB schema.

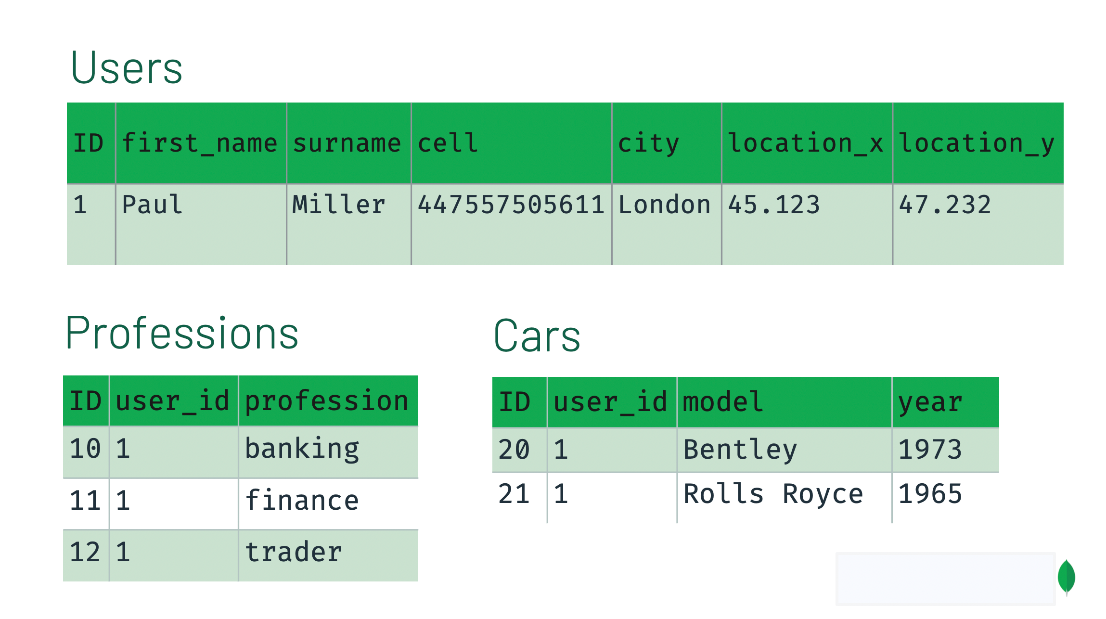


But in reality it’s a big



##### **Relational Schema Design**

When designing a relational schema, typically, developers model their schema independent of queries. They ask themselves, "What data do I have?" Then, by using prescribed approaches, they will (typically in 3NF). The idea of normalization is to split up your data into tables, so you don't duplicate data. Let's take a look at an example of how you would model some user data in a relational database.

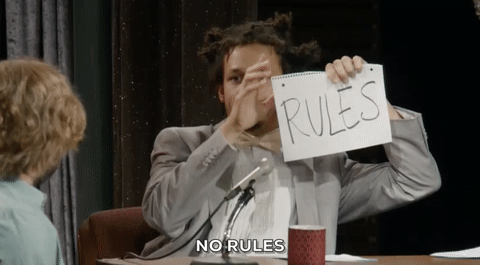


In this example, you can see that the user data is split into separate tables and it can be JOINED together using foreign keys in the user\_id column of the Professions and Cars table. Now, let's take a look at how we might model this same data in MongoDB.

##### **MongoDB Schema Design**

Now, MongoDB schema design works a lot differently than relational schema design. With MongoDB schema design, there is:

* No formal process
* No algorithms
* No rules



When you are designing your MongoDB schema design, the only thing that matters is that you design a schema that will work well for ***your*** application. Two different apps that use the same exact data might have very different schemas if the applications are used differently. When designing a schema, we want to take into consideration the following:

* Store the data
* Provide good query performance
* Require reasonable amount of hardware

Let's take a look at how we might model the relational User model in MongoDB.



You can see that instead of splitting our data up into separate collections or documents, we take advantage of MongoDB's document based design to embed data into arrays and objects within the User object. Now we can make one simple query to pull all that data together for our application.

<https://www.mongodb.com/developer/products/mongodb/mongodb-schema-design-best-practices/>