**How I learned Mongoose JS -** [Md. Shahriar Mehedi](https://medium.com/@sau.shahriar?source=post_page-----491c69db9c54--------------------------------) Feb 25, 2022

Today I am writing a blog about how I learned a new technology called mongoose which is a database framework for MongoDB.

Mongoose is an Object Data Modeling (ODM) library for MongoDB and Node.js. It manages relationships between data, provides schema validation, and is used to translate between objects in code and the representation of those objects in MongoDB.

Mongoose provides a straight-forward, schema-based solution to model your application data. It includes built-in type casting, validation, query building, business logic hooks, and more, out of the box. It’s basically a JavaScript layer on top of MongoDB and it adds some extra fancy things in that.

**Terminologies I have learned with mongoose:**

**Collections**

‘Collections’ in Mongoose are equivalent to tables in relational databases. They can hold multiple JSON documents.

**Documents**

‘Documents’ are equivalent to records or rows of data in SQL. While a SQL row can reference data in other tables, Mongoose documents usually combine that in a document.

**Fields**

‘Fields’ or attributes are similar to columns in a SQL table.

**Schema**

While MongoDB is schema-less, SQL defines a schema via the table definition. A Mongoose ‘schema’ is a document data structure (or shape of the document) that is enforced via the application layer.

**Models**

‘Models’ are higher-order constructors that take a schema and create an instance of a document equivalent to records in a relational database.

**The three main advantages of using Mongoose are:**

1. Mongoose JS provides an abstraction layer on top of MongoDB that eliminates the need to use named collections.
2. Models in Mongoose perform the bulk of the work of establishing up default values for document properties and validating data.
3. Functions may be attached to Models in Mongoose JS. This allows for the seamless incorporation of new functionality.