* **DataBase Excess using Node**

Now it is the time when we use to add our data and get our data instead of using file to save as cart.json and product.json

So we are using data base because it will work fast so instead of using file like json to read and write (**which will take a lot more time because we have to read whole file for particular data so**) we use data base

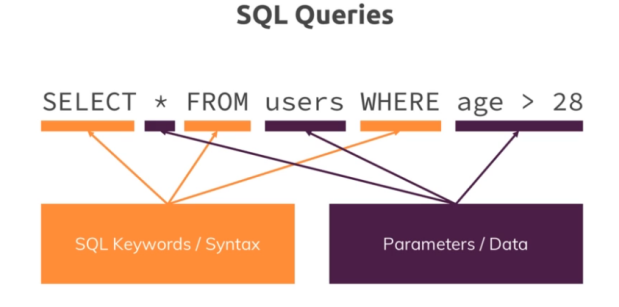
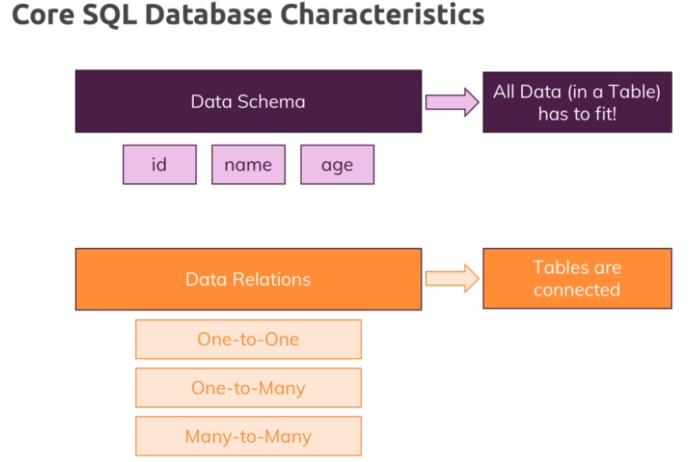
Data base are sql,nosql



In sql which is structured query language

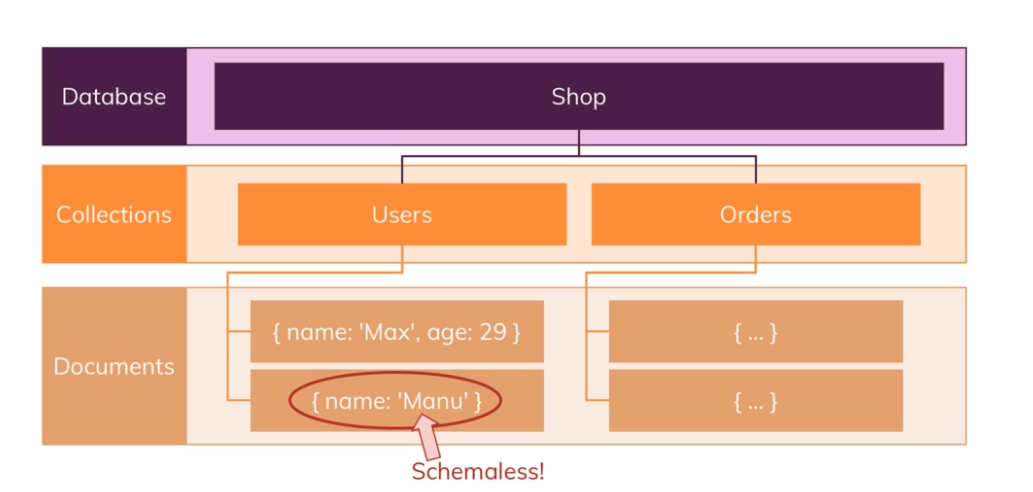
It provides a data in structured format in which we have data schema which tells us how the data look like and the data relation that is the relation between the multiple data lie sharing the same name

Queries :- queries are the commands which we use to interact with data base

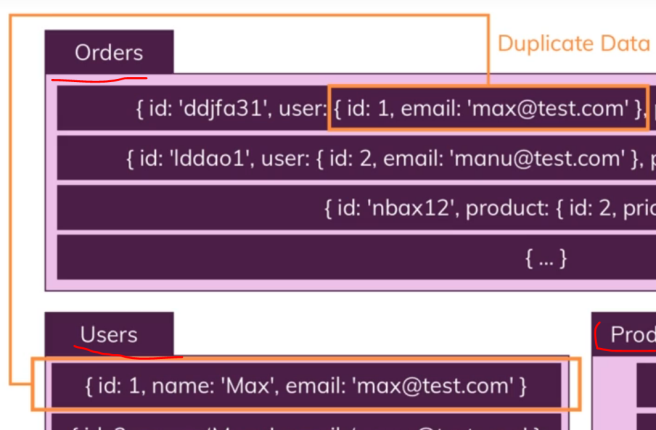
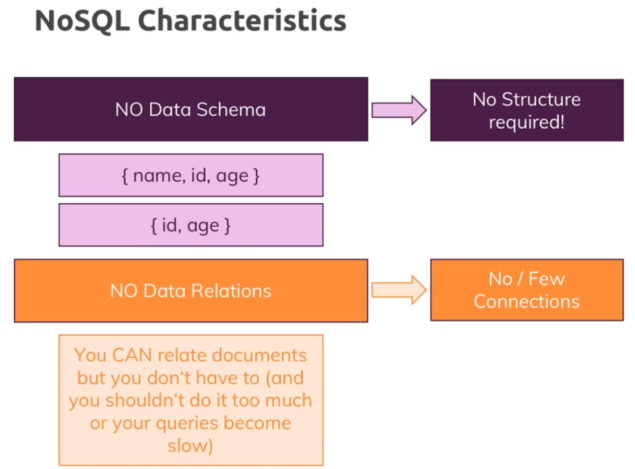


**noSql database**

so in noSql data base tables are called collection in which we stores our document which is not in particular structure in which we can add the random data and might be possible that we can miss some values like age etc.

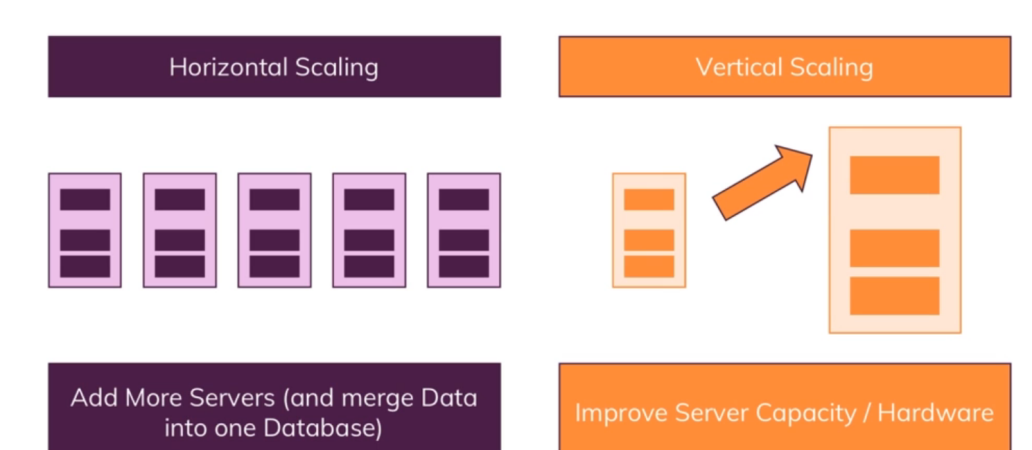


And we do not have any relation between the collection instead we duplicate our data every time for different collections which makes it very fast but increase the memory

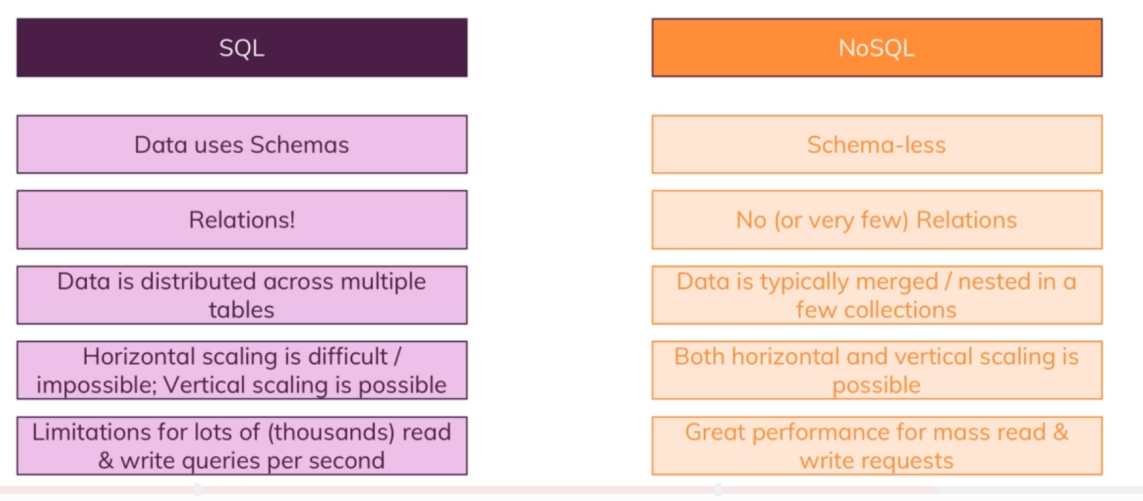
 

**Scaling in our database**

There is two types of scaling that is horizontal and vertical in horizontal we can add more or large number of server and for vertical scaling we just increase the cpu memory or increase cpu power



Which used which



* **Adding Sql to in our project**

So first we have to install sql (sql x86 and work bench) you have to open the sql and add schema to it

Now to add data base we have to first install –save mysql2 in our dependency



Now after that we have to require mysql and pass our database which we created so we can use the database to get or post the data

So for that we have to add the database connection file in our util folder

// connecting dataBase in the project

const mysql = require('mysql2');

const pool = mysql.createPool({

    host: 'localhost',

    user: 'root',

    database: 'product-data',

    password: 'Abcd@1234'

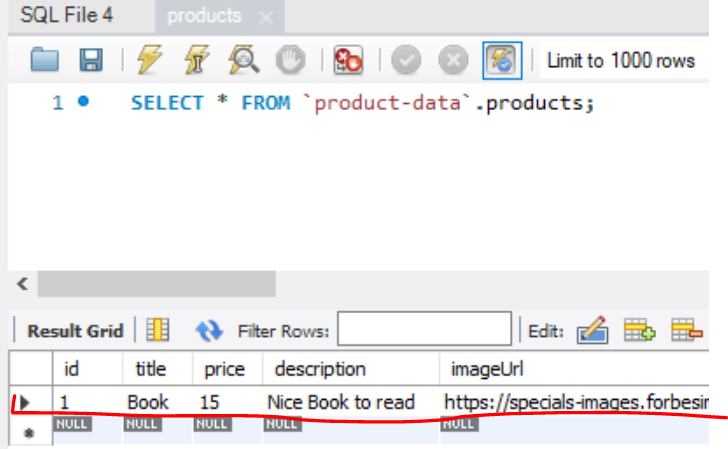
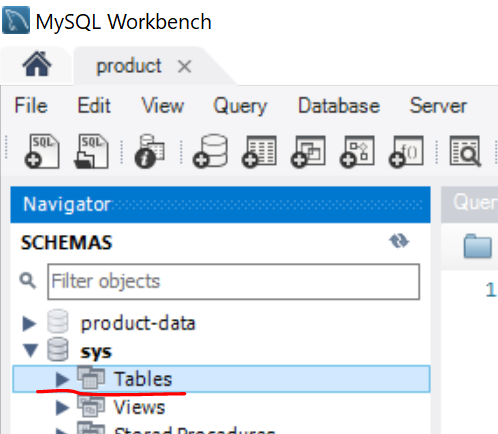
});

module.exports = pool.promise();

it is important that hostname, user name,database name, password should be equal to our sql workspace that we are created in sql , so the required sql can interact with the same file

**connecting database to our code just for testing.**

here product-data is the schema or our database name and from that we can create our table an can file the data inside it. So that we can retrieve the data



To get the database connectivity we first have to connect it and use it inside our app.js or main js file which is listening our data ad we can do it by requiring the exports of pool db

const db = require('./util/dataBase');

because the above exports giving us a promise so we can use then ad catch after promise

db.execute('SELECT \* FROM products')

.then(result => {

    console.log(result[0],result[1]);

})

.catch(err => {

    console.log(err);

})

So above we are just executing the command in the mysql by vs code and then we use result which holds the data and when we console log( result) so result[0] will give our data and result[1] give meta data with it.

**After testing lets jump to real connectivity**

So for real time connectivity we have remove all the function in our model which we are using previously because we are getting and sending data from the file that we have created in our pc so we are doing read and write operation all the time

So now we do not need to read and write instead we are using database in our models

So to attach the shop and product file with my sql we have to use the update our controller so that the controller can interact with the models and view by mysql data

So for that we have to use my sql command inside our models we can select the all file or we can select some specific by id

const db = require('../util/dataBase');

const Cart = require('./cart');

module.exports = class Product {

  constructor(id, title, imageUrl, description, price) {

    this.id = id;

    this.title = title;

    this.imageUrl = imageUrl;

    this.description = description;

    this.price = price;

  }

  // here we have to provide specific fields for (?,?,?,?) as an values in form of array because we are writing in mysql by code

  save() {

    return db.execute('INSERT INTO products (title,price,imageUrl,description) VALUES(?,?,?,?)',

    [this.title,this.price,this.imageUrl,this.description]);

  }

  static deleteById(id) {

  }

  // it will get all the data

  static fetchAll() {

    return db.execute('SELECT \* FROM products');

  }

  // it will select specific data

  static findById(id) {

    return db.execute('SELECT \* FROM products where products.id = ?', [id]);

  }

};

Now we have to change our controller code for all the files because we changed our way of taking and sending data

exports.getProducts = (req, res, next) => {

  Product.fetchAll()

  .then(([data,metaData]) => {

      res.render('shop/product-list', {

        prods: data,

        pageTitle: 'All Products',

        path: '/products'

      });

  })

  .catch(err => console.log(err))

};

// the data is present in the 0 index

exports.getProduct = (req, res, next) => {

  const prodId = req.params.productId;

  Product.findById(prodId)

  .then(([product]) => {

    res.render('shop/product-detail', {

      product: product[0],

      pageTitle: product.title,

      path: '/products'

    });

  })

  .catch(err => console.log(err))

};