



# Sequelize

Node | Express | Sequelize | MySQL

# Example 01 SQL Database

## Users

id	email	name
1	basit@gmail.com	Basit Hussain
2	ammar@test.com	Ammar Ahmed
3	ali@xyz.com	Ali

{id: 1, name: 'Afzal'}

{id: 1, title: 'Book'}

{1, {id: 1, name: 'Afzal'}, {id: 1, title: 'Book'}}

## Products

id	title	price
1	Laptop	\$1200
2	Smart Phone	\$1000
3	Book	\$400

## Orders

id	user_id	product_id
1	2	2
2	1	2
3	3	3

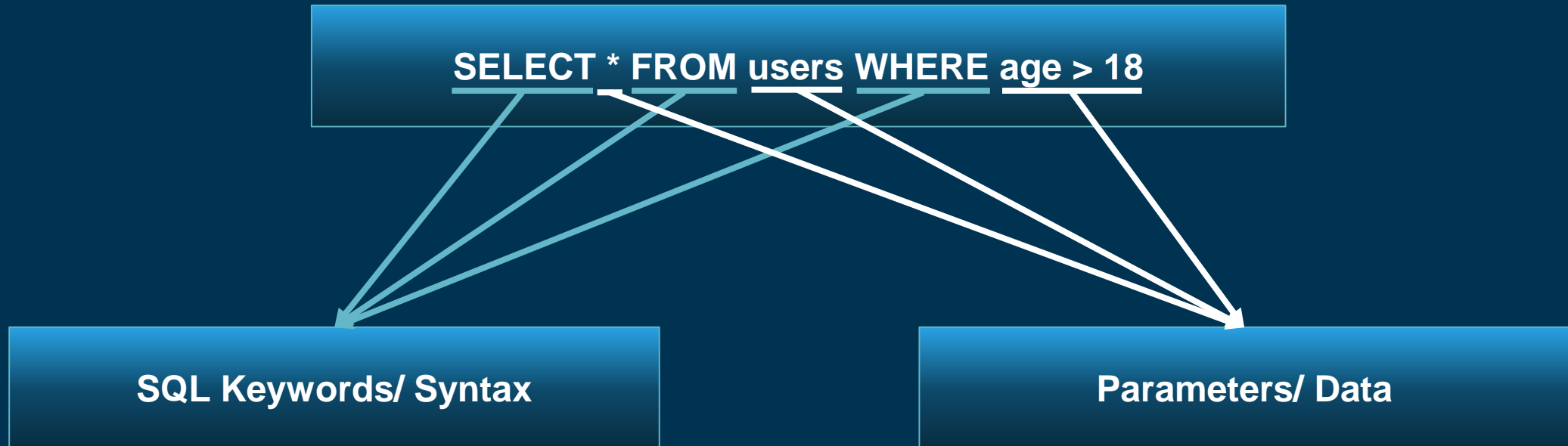
# Data Relations

**One to One**

**One to Many**

**Many to Many**

# SQL Queries



# What is Sequelize

An **O**bject-**R**elational **M**apping Library

User
<ul style="list-style-type: none"><li>• Name</li><li>• Email</li><li>• Password</li></ul>

Mapped

Users			
id	name	email	Password
1	Basit Hussain	basit@gmail.com	123

```
INSERT INTO users VALUES (1, 'Basit Hussain', 'basit@gmail.com', '123')
```

# Sequelize Core Concepts

**Models**

e.g. User, Products

**Instances**

```
const user = User.build()
```

**Queries**

```
User.findAll()
```

**Associations**

```
User.hasMany(Products)
```

# Get to Ready to start the Code...

```
npm i express sequelize cors mysql2
```

# CORS

- Cross Origin Resource Sharing (CORS)



# Create Server

```
1  const express = require('express');
2  const cors = require('cors');
3
4  const app = express();
5
6  const corsOptions = {
7    origin: 'http://localhost:8080',
8  };
9
10 // middlewares
11
12 app.use(cors(corsOptions));
13 app.use(express.json());
14 app.use(express.urlencoded({extended: true}));
15
16 // testing api
17 app.get('/', (req, res) => {
18   res.json({message: 'hello'})
19 });
20
21 // port
22
23 const PORT = process.env.PORT || 8080;
24
25 // server
26 app.listen(PORT, () => {
27   console.log(`server is running on PORT ${PORT}`);
28 })
29
```

# Middleware

```
1 // global middleware
2 app.use(cors(corsOptions));
3 app.use(express.json());
4 app.use(express.urlencoded({extended: true}));
5 app.use(middleware);
6
7 // api
8 app.get('/', (req, res) => {
9     console.log('api testing');
10    res.json({
11        name: 'Basit',
12        email: 'basit@gmail.com',
13    })
14 })
15
16 app.get('/contact', (req, res) => {
17     console.log('contact page');
18     res.send('Contact Page')
19 })
20
21
22 function middleware(req,res,next) {
23     console.log('middleware');
24     next()
25 }
```

# Middleware

```
1 // global middleware
2 app.use(cors(corsOptions));
3 app.use(express.json());
4 app.use(express.urlencoded({extended: true}));
5
6 // api
7 app.get('/', (req, res) => {
8   console.log('api testing');
9   res.json({
10     name: 'Basit',
11     email: 'basit@gmail.com',
12   })
13 })
14
15 app.get('/contact', (req, res, next) => {
16   console.log('contact page');
17   res.send('Contact Page');
18   next();
19 })
20
21 app.use(middleware);
22
23
24
25 function middleware(req, res, next) {
26   console.log('middleware');
27   next()
28 }
```

# Middleware

```
1 app.get('/contact', auth, (req, res, next) => {
2     console.log('contact page');
3     res.send('Contact Page');
4 })
5
6 function auth(req, res, next) {
7     if (req.query.admin) {
8         next();
9     } else {
10         res.send('Auth Required')
11     }
12 }
```

# Middleware Data Pass

```
1 const corsOptions = {
2   origin: 'http://localhost:8080'
3 };
4
5 // global middleware
6 app.use(cors(corsOptions));
7 app.use(express.json());
8 app.use(express.urlencoded({extended: true}));
9 app.use(middleware);
10
11
12 // api
13 app.get('/', (req, res) => {
14   console.log('api testing');
15   res.json({
16     name: 'Basit',
17     email: 'basit@gmail.com',
18   })
19 })
20
21 app.get('/contact', auth, (req, res, next) => {
22   console.log(`Auth data ${req.admin}`);
23   res.send('Contact Page');
24 })
25
26 function middleware(req, res, next) {
27   console.log('middleware');
28   console.log(req.originalUrl);
29   next()
30 }
31
32 function auth(req, res, next) {
33   if (req.query.admin === 'true') {
34     req.admin = true
35     next();
36   } else {
37     res.send('Auth Required')
38   }
39 }
```

# Middleware Data Pass

```
1 // api
2 app.get('/', (req, res) => {
3   console.log('api testing');
4   res.json({
5     name: 'Basit',
6     email: 'basit@gmail.com',
7   })
8 })
9
10 app.get('/contact', auth, (req, res, next) => {
11   console.log(`Auth data ${req.admin}`);
12   res.send('Contact Page');
13 })
14
15 function middleware(req, res, next) {
16   console.log('middleware');
17   console.log(req.originalUrl);
18   next()
19 }
20
21 function auth(req, res, next) {
22   if (req.query.admin === 'true') {
23     req.admin = true
24     next();
25     return
26   }
27
28   res.send('Auth Required')
29 }
```

# Config.js

```
1  module.exports = {  
2    HOST: 'localhost',  
3    USER: 'root',  
4    PASSWORD: '',  
5    DB: 'seq_basics',  
6    dialect: 'mysql',  
7  
8    pool: {  
9      max: 5,  
10     min: 0,  
11     acquire: 30000,  
12     idle: 10000,  
13   }  
14 }  
15
```

# model/index.js

```
1 const dbConfig = require('../config/dbConfig');
2 const {Sequelize, DataTypes} = require('sequelize');
3 const sequelize = new Sequelize(
4   dbConfig.DB,
5   dbConfig.USER,
6   dbConfig.PASSWORD,
7   {
8     host: dbConfig.HOST,
9     dialect: dbConfig.dialect,
10    operatorAliases: false,
11    pool: {
12      max: dbConfig.pool.max,
13      min: dbConfig.pool.min,
14      acquire: dbConfig.pool.acquire,
15      idle: dbConfig.pool.idle,
16    }
17  }
18 );
19
20 sequelize.authenticate()
21 .then(() => {
22   console.log('Connected');
23 }).catch((err) => {
24   console.log(err);
25 });
26
27 const db = {}
28 db.Sequelize = Sequelize;
29 db.sequelize = sequelize;
30
31 db.products = require('./productModel.js')(sequelize, DataTypes);
32 db.reviews = require('./reviewModel.js')(sequelize, DataTypes);
33
34 db.sequelize.sync({force: false})
35 .then(() => {
36   console.log('Drop and re-sync db.');
```



# model/ productModel.js

```
1 module.exports = (sequelize, DataTypes) => {  
2   const Product = sequelize.define("product", {  
3     title: {  
4       type: DataTypes.STRING,  
5       allowNull: false,  
6     },  
7     price: {  
8       type: DataTypes.INTEGER,  
9     },  
10    description: {  
11      type: DataTypes.TEXT,  
12    },  
13    published: {  
14      type: DataTypes.BOOLEAN,  
15    },  
16  });  
17  return Product;  
18 };  
19
```


# Controller/ ProductController.js

```
1 const db = require("../models");
2
3 // create main model
4 const Product = db.products;
5
6 // create product
7 const addProduct = async (req, res) => {
8   // validate request
9   if (!req.body.title) {
10     res.status(400).send({
11       message: "Content can not be empty!",
12     });
13     return;
14   }
15
16   // create a product
17   let info = {
18     title: req.body.title,
19     price: req.body.price,
20     description: req.body.description,
21     published: req.body.published ? req.body.published : false,
22   };
23
24   // save Product in the database
25   try {
26     const product = await Product.create(info);
27     res.status(200).send(product);
28     console.log(product);
29   } catch (err) {
30     res.status(500).send({
31       message: err.message || "Error occurred while creating the Product",
32     });
33   }
34 };
```

# Controller/ ProductController.js

```
1 // get all products
2 const getAllProducts = async (req, res) => {
3   // let products = await Product.findAll({
4   //   attributes: [
5   //     'title',
6   //     'price'
7   //   ]
8   // });
9
10  let products = await Product.findAll({});
11  res.status(200).send(products);
12  console.log(products);
13  };
```

# Controller/ ProductController.js



```
1 // get single products
2 const getSingleProduct = async (req, res) => {
3   let id = req.params.id;
4   let product = await Product.findOne({ where: { id: id } });
5   res.status(200).send(product);
6 };
```

# Controller/ ProductController.js



```
1 // update a product
2 const updateProduct = async (req, res) => {
3   let id = req.params.id;
4   const product = await Product.update(req.body, { where: { id: id } });
5   res.status(200).send('Product is Updated');
6 };
```

# Controller/ ProductController.js



```
1 // delete a product
2 const deleteProduct = async (req, res) => {
3   let id = req.params.id;
4   await Product.destroy({ where: { id: id } });
5   res.status(200).send("Product is deleted");
6 };
```

# Controller/ ProductController.js



```
1 // get published product
2 const getPublishedProduct = async (req, res) => {
3   const products = await Product.findAll({ where: { published: true } });
4   res.status(200).send(products);
5 };
```

# Controller/ ProductController.js



```
1 module.exports = {  
2   addProduct,  
3   getAllProducts,  
4   getPublishedProduct,  
5   getSingleProduct,  
6   updateProduct,  
7   deleteProduct,  
8 };;
```



# routes/productRoutes.js

```
1  const productController = require("../controllers/productController.js");
2
3  const router = require("express").Router();
4
5  router.post("/addProduct", productController.addProduct);
6  router.get("/allProducts", productController.getAllProducts);
7  router.get("/published", productController.getPublishedProduct);
8  router.get("/:id", productController.getSingleProduct);
9  router.put("/:id", productController.updateProduct);
10 router.delete("/:id", productController.deleteProduct);
11
12 module.exports = router;
13
```

# Server.js



```
1
2 // routers
3 const router = require('./routes/productRouter');
4 app.use('/api/products', router);
```

# Content Title



Caption01 appears  
here



Caption02 appears  
here



Caption03 appears  
here



Caption04 appears  
here



Caption05 appears  
here

# Content Title 03



Caption01 appears here

Caption03 appears here

Caption04 appears here

# Content Title 04



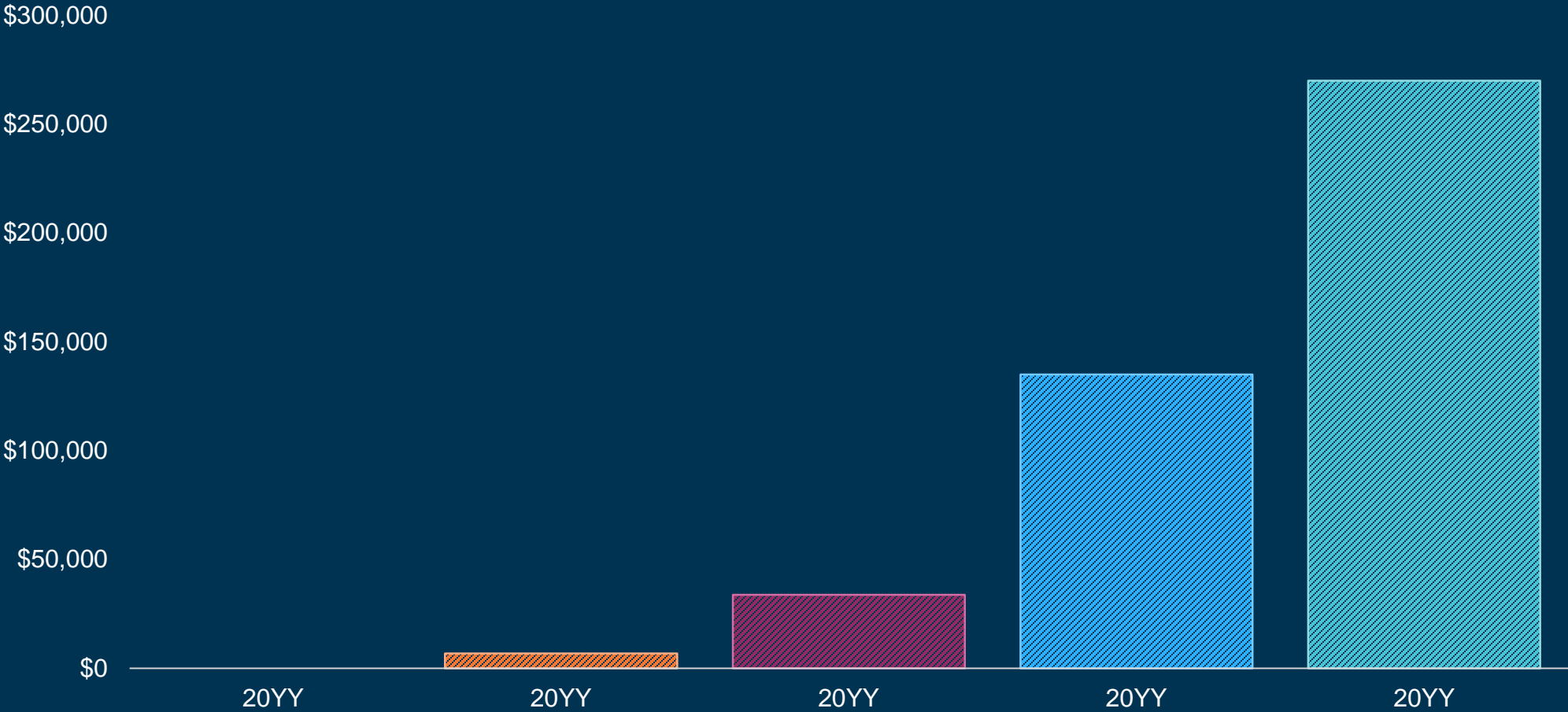
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Maecenas porttitor congue massa. Fusce posuere, magna sed pulvinar ultricies, purus lectus malesuada libero, sit amet commodo magna eros quis urna. Nunc viverra imperdiet enim. Fusce est. Vivamus a tellus.

Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Proin pharetra nonummy pede. Mauris et orci.

# Table

Title	Title	Title	Title

# Chart





Quote appears here  
Lorem ipsum dolor sit amet,  
consectetuer adipiscing elit.”  
- Author



The background is a dark blue gradient. A diagonal line runs from the bottom-left towards the top-right. To the left of this line is a lighter blue area. To the right is a darker blue area. A thin, light blue striped band follows the diagonal line.

**Thank You 1**



**Thank You 2**

Customize this Template

# Template Editing Instructions and Feedback