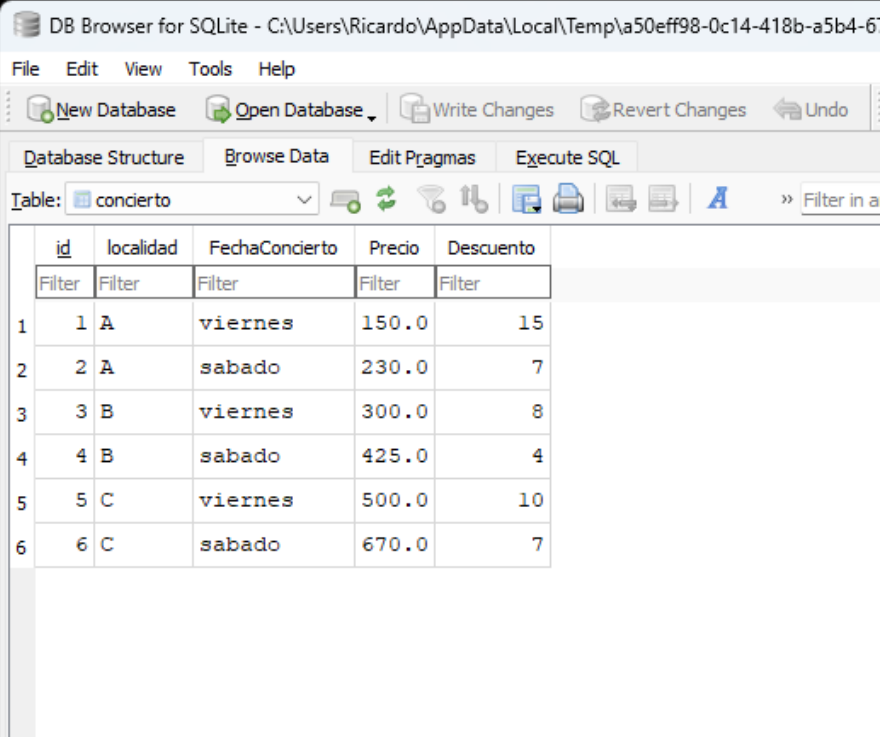


Primero crearemos la base de datos con las columnas localidad, FechaConcierto, Precio y Descuento y la llenaremos con registros, esta base de datos sera la que conectaremos al proyecto y se llamará conciertos.sqlite



DB Browser for SQLite - C:\Users\Ricardo\AppData\Local\Temp\a50eff98-0c14-418b-a5b4-6

File Edit View Tools Help

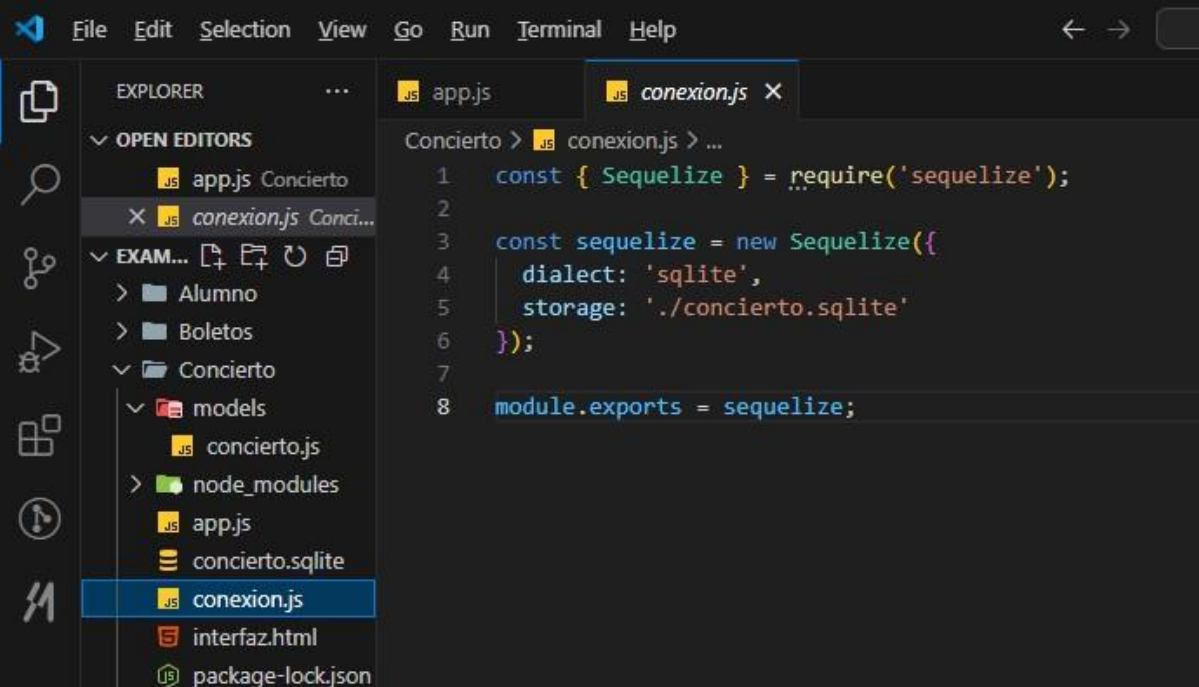
New Database Open Database Write Changes Revert Changes Undo

Database Structure Browse Data Edit Pragma Execute SQL

Table: concierto

	id	localidad	FechaConcierto	Precio	Descuento
	Filter	Filter	Filter	Filter	Filter
1	1	A	viernes	150.0	15
2	2	A	sabado	230.0	7
3	3	B	viernes	300.0	8
4	4	B	sabado	425.0	4
5	5	C	viernes	500.0	10
6	6	C	sabado	670.0	7

Después creamos la conexión de la base de datos al proyecto con sequelize como se ha hecho en los proyectos anteriores



File Edit Selection View Go Run Terminal Help

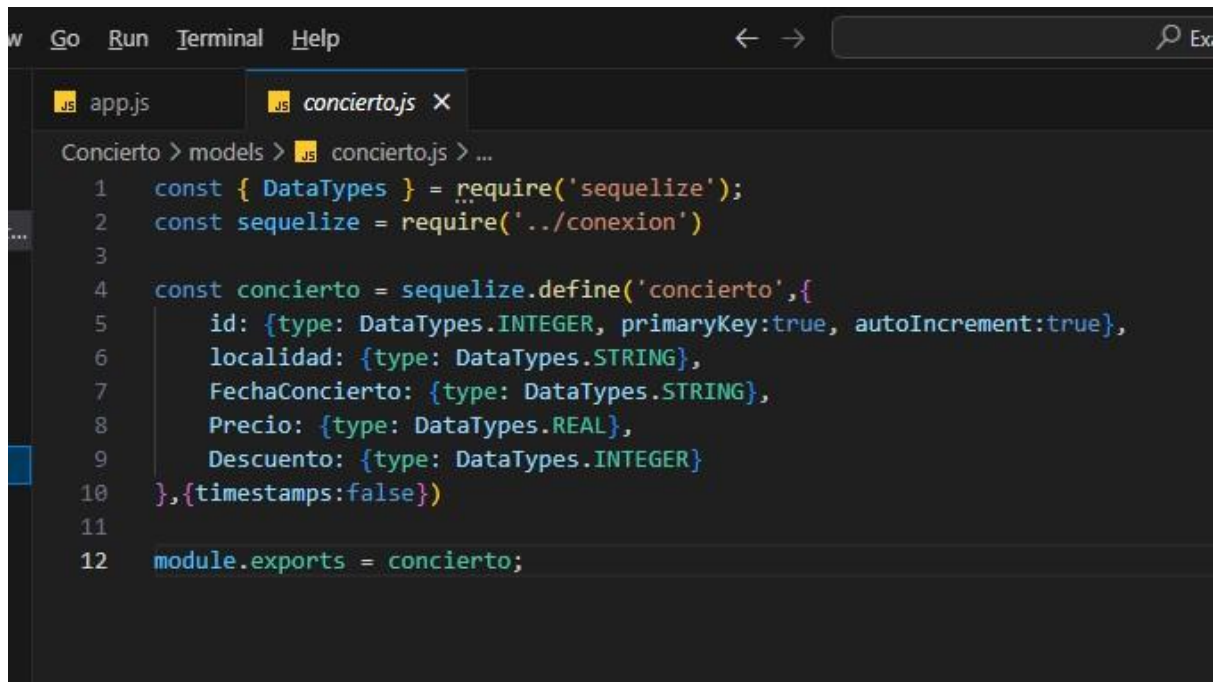
EXPLORER

- OPEN EDITORS
 - app.js Concierto
 - conexion.js Conci...
- EXAM...
 - Alumno
 - Boletos
 - Concierto
 - models
 - concierto.js
 - node_modules
 - app.js
 - concierto.sqlite
 - conexion.js
 - interfaz.html
 - package-lock.json

Concierto > conexion.js > ...

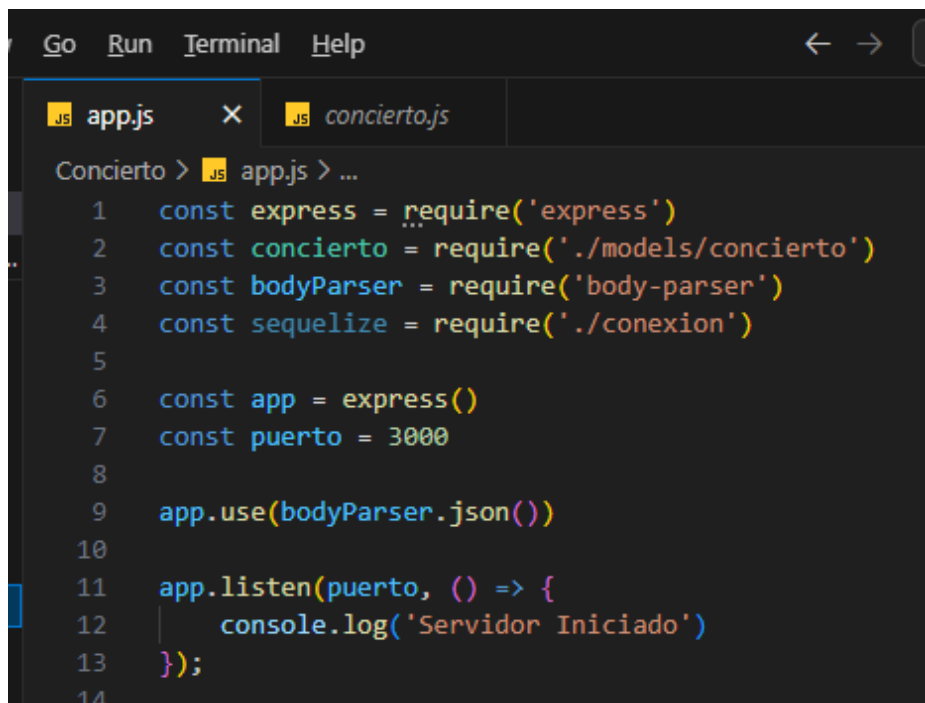
```
1 const { Sequelize } = require('sequelize');
2
3 const sequelize = new Sequelize({
4   dialect: 'sqlite',
5   storage: './concierto.sqlite'
6 });
7
8 module.exports = sequelize;
```

luego crearemos el modelo concierto con las columnas y los parámetros de la base de datos para que pueda funcionar correctamente



```
Concierto > models > JS concierto.js > ...
1  const { DataTypes } = require('sequelize');
2  const sequelize = require('../conexion')
3
4  const concierto = sequelize.define('concierto',{
5      id: {type: DataTypes.INTEGER, primaryKey:true, autoIncrement:true},
6      localidad: {type: DataTypes.STRING},
7      FechaConcierto: {type: DataTypes.STRING},
8      Precio: {type: DataTypes.REAL},
9      Descuento: {type: DataTypes.INTEGER}
10 },{timestamps:false})
11
12 module.exports = concierto;
```

En el app principal importamos los módulos y definiremos las constantes, en esta ocasión se agregó body-parser, definiremos a qué puerto se iniciara el servidor y ponemos un mensaje de que el servidor se inicializa



```
Concierto > JS app.js > ...
1  const express = require('express')
2  const concierto = require('./models/concierto')
3  const bodyParser = require('body-parser')
4  const sequelize = require('../conexion')
5
6  const app = express()
7  const puerto = 3000
8
9  app.use(bodyParser.json())
10
11 app.listen(puerto, () => {
12     console.log('Servidor Iniciado')
13 });
14
```

en el body de la API definiremos la ruta para que funcione el servidor web y haremos la validación de los datos en la base de datos y arrojaremos un mensaje si hubo un error y después haremos el calculo del descuento de ser necesario

```
10
11 app.listen(puerto, () => {
12   console.log('Servidor Iniciado')
13 });
14
15 app.post('/concierto', async (req, res) => {
16   const { localidad, FechaConcierto, estudiante } = req.body;
17
18   const data = await concierto.findOne({
19     where: { localidad, FechaConcierto }
20   });
21
22   if (!data) {
23     return res.status(404).json({ error: 'No se encontró esa localidad y fecha' });
24   }
25
26   let { Precio, Descuento } = data;
27
28   if (estudiante === true && (FechaConcierto === 'viernes' || FechaConcierto === 'sabado')) {
29     let precioConDescuento = Precio - (Precio * Descuento / 100);
30     res.send({ Precio: precioConDescuento, localidad, FechaConcierto });
31   } else {
32     res.send({ Precio, localidad, FechaConcierto });
33   }
34 });
```