

Mongo db

Práctica del tema 8 - DSI

MongoDB & NodeJS

- La idea de la práctica es confeccionar un conjunto de facturas de prueba para una hipotética aplicación.
- Partiendo de un archivo de empleados y una base de datos (en formato CSV) de productos debemos introducir en MongoDB una colección (lo que vendría a ser una tabla en el sentido relacional) con la lista de productos y otra con 1.000 facturas.
- La estructura de los documentos en la colección productos contiene solamente el producto y el precio (échale un vistazo al .csv que te paso), las facturas tienen el siguiente aspecto:

Datos de origen

- El archivo CSV de productos puedes encontrarlo en el campus virtual y en la siguiente dirección:

[Archivo de productos \(CSV\)](#)

- El listado de empleados puedes encontrarlo en el campus virtual y en la siguiente dirección:

[Listado de empleados](#)

Instalación y ejecución

- Debéis instalar MongoDB y ejecutar **mongod** para que se ejecute el demonio y podáis conectaros al SGBD
- Para acceder a la consola es con **mongo**
- Para acceder desde nodejs hay que instalar el módulo correspondiente con la instrucción:

```
npm install mongo
```

```
1 var MongoClient = require('mongodb').MongoClient;
2
3 // Conectar a la base de datos
4 MongoClient.connect("mongodb://localhost:27017/", function(err, database)
5 {
6     if(err) { return console.dir(err); }
7
8     const MyDB = database.db('test');
9     var ColeccionProductos = MyDB.collection('Productos');
10    var FileProductos = fs.readFileSync('Salida.csv', 'utf8');
11    var FileEmpleados = fs.readFileSync('Nombres.txt', 'utf8');
12
13    // Bucle para todos los productos, leer cada linea del fichero
14    // recuerda la primera linea contiene los encabezados.
15    //{
16        // Creo el objeto javascript en formato JSON y lo inserto
17        var Tupla = {'Producto':Valores[0], 'Precio':Valores[1]};
18        ColeccionProductos.insert(Tupla);
19    //}
20 });
21
```

Insertando facturas

- Debes realizar un bucle similar para insertar 1.000 facturas.
- Escoge un Empleado al azar e insértalo en la factura.
- Escoge un número de productos que iran en la factura (de 1 a 5).
- Escoge un número que será la cantidad de productos del mismo tipo (por ejemplo 5 bricks de leche).
- Calcula los totales y el IGIC.
- Recuerda el “tipado de patos” de javascript que te permite añadir las propiedades a posteriori, así puedes sumar los totales y añadirlos a la factura.
- Pon atención a cómo generar un array de objetos.

```
1  for (var i = 0; i < 1000; i++)
2  {
3      var Factura = { 'Empleado': DameEmpleado(Empleado) };
4      var Detalle = [];
5      for (var P = 1; P <= Aleatorio(5); P++) // De 1 a 5 productos
6      {
7          Cantidad = Aleatorio(10); // Cantidad aleatoria del mismo articulo
8
9          // Seleccionar aleatoriamente un producto de la lista
10         // Calcular Precio (Cantidad * Importe), IGIC, Total
11         Detalle.push({ 'Cantidad': Cantidad, 'Producto': Vector[0]
12             , 'Importe': Vector[1], 'IGIC': IGIC, 'Total': Total});
13
14         // Acumular Totales
15     }
16     Factura.SubTotal = SubTotal; // "Tipado de patos"
17     Factura.IGIC = TotalIGIC;
18     Factura.Total = TotalFactura;
19     Factura.Detalle = Detalle;
20     ColeccionFacturas.insert(Factura);
21 }
```


Finalizando

- Ahora que tienes datos, saca el sumatorio de los totales de las facturas, desde la consola o con tu propio programa.
- Quizás te sea útil esta documentación:

[\\$sum](#)

- Abre una consola, realiza búsquedas con find() y crea algún índice ¿notas diferencias?