María Prados

AGGREGATION

En este trabajo vamos a utilizar el operador **Aggregate** para hacer algunas consultas operacionales con nuestra búsqueda, en este caso la contabilidad de proveedor de pizzerias, lo he hecho con productos finales para no tener que meterme en materias primas. Y poder usar también los precios de clientes, esto no seria del todo fiel a la realidad, puesto que habría que hacer varias colecciones, mínimo 2, una para los proveedores y otra para los clientes de la pizzería.

Estructura

La estructura de nuestra colección incluye un id, y varios campos numéricos de precio, cantidad, beneficio y rentabilidad, un campo en formato fecha, tres campos tipo string con los nombres de los empleados y clientes y un booleano.

Para el campo **"beneficios_r**ate", he utilizado la siguiente fórmula mediante reglas de tres he relacionado el precio de venta cliente (final) con el precio de coste. Calculado en porcentaje.

Y para el campo "**rentabilidad_rate**", de la misma manera he relacionado el precio de venta de proveedor (intermedio), con el precio de venta de cliente, para así obtener la rentabilidad neta, obtenida en porcentaje.

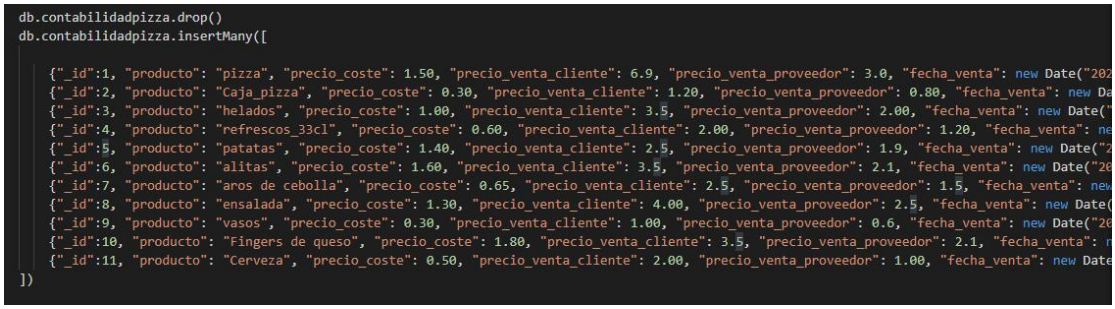
Estos cálculos son sin IVA. Luego he incluido un porcentaje de IVA 4% para productos de alimentación y 16% para bebidas y artículos desechables. Así como un descuento para grandes pedidos en el caso de que sean mayoristas (un 5% a partir de 500 unidades y un 10% a partir de 1000 unidades).

```
db.contabilidadpizza.drop()
db.contabilidadpizza.insertOne(

    { "_id" : 1,
      "producto": "pizza",
      "precio_coste": 1.50,
      "precio_venta_cliente": 6.9,
      "precio_venta_proveedor": 3.0,
      "fecha_venta": new Date("2021-12-19"),
      "cantidad": 600,
      "cliente": "Pizzería Deliciosa",
      "empleado_vendedor": "Luis Santos",
      "beneficios_rate": 460,
      "rentabilidad_rate": 56.5,
      "IVA": 10,
      "Mayorista": true,
      "Dto. Mayorista": 5
    }
)
```

Inserts

Mi colección esta formada por 11 inserts, cada uno con su id único de producto.



```
db.contabilidadpizza.drop()
db.contabilidadpizza.insertMany([

    {"_id":1, "producto": "pizza", "precio_coste": 1.50, "precio_venta_cliente": 6.9, "precio_venta_proveedor": 3.0, "fecha_venta": new Date("202
    {"_id":2, "producto": "Caja_pizza", "precio_coste": 0.30, "precio_venta_cliente": 1.20, "precio_venta_proveedor": 0.80, "fecha_venta": new Da
    {"_id":3, "producto": "helados", "precio_coste": 1.00, "precio_venta_cliente": 3.5, "precio_venta_proveedor": 2.00, "fecha_venta": new Date("
    {"_id":4, "producto": "refrescos_33cl", "precio_coste": 0.60, "precio_venta_cliente": 2.00, "precio_venta_proveedor": 1.20, "fecha_venta": ne
    {"_id":5, "producto": "patatas", "precio_coste": 1.40, "precio_venta_cliente": 2.5, "precio_venta_proveedor": 1.9, "fecha_venta": new Date("2
    {"_id":6, "producto": "alitas", "precio_coste": 1.60, "precio_venta_cliente": 3.5, "precio_venta_proveedor": 2.1, "fecha_venta": new Date("20
    {"_id":7, "producto": "aros de cebolla", "precio_coste": 0.65, "precio_venta_cliente": 2.5, "precio_venta_proveedor": 1.5, "fecha_venta": new
    {"_id":8, "producto": "ensalada", "precio_coste": 1.30, "precio_venta_cliente": 4.00, "precio_venta_proveedor": 2.5, "fecha_venta": new Date(
    {"_id":9, "producto": "vasos", "precio_coste": 0.30, "precio_venta_cliente": 1.00, "precio_venta_proveedor": 0.6, "fecha_venta": new Date("20
    {"_id":10, "producto": "Fingers de queso", "precio_coste": 1.80, "precio_venta_cliente": 3.5, "precio_venta_proveedor": 2.1, "fecha_venta": n
    {"_id":11, "producto": "Cerveza", "precio_coste": 0.50, "precio_venta_cliente": 2.00, "precio_venta_proveedor": 1.00, "fecha_venta": new Date
])
```

Consultas

Para las búsquedas he usado los distintos operadores mencionados en el enunciado del trabajo **$match y $group**, para filtrar y agrupar la búsqueda.

Para las operaciones he incluido **$sum, $multiply** para los sumatorios de precios y multiplicar por cantidades. También para el calculo del IVA.

$avg para calcular la cantidad media.
$max para calcular los precios máximos.
$divide para un supuesto en el que dividir un presupuesto estimado.
$sustract para restar el descuento de mayorista
$round para redondear cantidades.

Ejemplos:

```
/*Calcular la media de productos vendidos por Pizza Delivery y su precio medio de venta de cliente*/
    db.contabilidadpizza.aggregate([

    {$match: {cliente: "Pizza Delivery"}},
    { $group: {
            _id: "$cliente",
            mediacantidad: {$avg:"$cantidad"},
            mediaprecio: {$avg: "$precio_venta_cliente"},
            numeroGrupo:{$sum:1}}}
    ])
```

```
/*Calcular las ventas totales particulares y su total de ventas*/

    db.contabilidadpizza.aggregate([
    { $match: { mayorista: "false" } },
    { $group: { _id: "$producto",
                sumQuantity: { $sum: "$cantidad" },
                venta_producto: {$multiply: ["precio_venta_cliente", "cantidad"]}}},
    { $group: {_id: null,  total: {$sum: "$sumQuantity"}}},
    {$project: {Producto: "_id",
                totalvProducto_No_Mayorista: $venta_producto}}
    ])
```

```
/*Obtener el precio de venta total, el iva y el precio total con iva en el año 2021*/

    db.contabilidadpizza.aggregate (
    [
        {$match: {$expr: {$lt: [{$year: "fecha_venta"}, 2022] }}},
    {$group:
        {
         _id: {$year: "$precio_venta"},
         venta_total: {$sum: {$multiply: ["precio_venta_cliente", "cantidad"]}}
        }
    },
    {$project: {
        año: "_id",
        _id: 0,
        totalventa: "$venta_total",
        IVA: {$multiply: ["$venta_total", 0.10]},
        TotalvIVA: {$sum: ["$venta_total", {$multiply: ["$venta_total", 0.10]}]},
        totalRedondeado: {$round: [{$sum: ["$venta_total", {$multiply: ["$venta_total", 0.10]}]}, 0]}
    }}
    ])
```