Práctica MongoDB

Aggregation Framework en MongoDB

1. Realizar una consulta que devuelva la siguiente información: Región y cantidad total de productos vendidos a clientes de esa Región.

```
db.facturas.aggregate([
  { $unwind: "$item" },
  {
    $group: {
      _id: "$cliente.region",
      productosVendidos: { $sum: "$item.cantidad" },
    },
  },
  {
    $project: {
      _id: 0,
      region: "$_id",
      productosVendidos: "$productosVendidos",
    },
  },
]);
```

```
[
  {
    "region": "NEA",
    "productosVendidos": 420
  },
  {
    "region": "CABA",
    "productosVendidos": 282
  },
  {
    "region": "NOA",
    "productosVendidos": 14
  },
  {
    "region": "CENTRO",
    "productosVendidos": 180
  }
]
```

2. Basado en la consulta del punto 1, mostrar sólo la región que tenga el menor ingreso.

```
db.facturas.aggregate([
  { $unwind: "$item" },
  {
    $group: {
      _id: "$cliente.region",
      productosVendidos: { $sum: "$item.cantidad" },
      totalVentas: { $sum: { $multiply: ["$item.cantidad",
"$item.precio"] } },
    },
  },
  { $sort: { totalVentas: 1 } },
  { $limit: 1 },
  {
    $project: {
      _id: 0,
      region: "$_id",
      productosVendidos: "$productosVendidos",
    },
  },
]);
```

```
[
  {
    "region": "NOA",
    "productosVendidos": 14
  }
]
```

3. Basado en la consulta del punto 1, mostrar sólo las regiones que tengan una cantidad de productos vendidos superior a 10000.

```
db.facturas.aggregate([
  { $unwind: "$item" },
  {
    $group: {
      _id: "$cliente.region",
      productosVendidos: { $sum: "$item.cantidad" },
      totalVentas: { $sum: { $multiply: ["$item.cantidad",
"$item.precio"] } },
    },
  },
  { $match: { totalVentas: { $gt: 10000 } } },
  { $sort: { totalVentas: 1 } },
  {
    $project: {
      _id: 0,
      region: "$_id",
      productosVendidos: "$productosVendidos",
      totalVentas: "$totalVentas",
    },
  },
]);
```

```
[
  {
    "region": "CENTRO",
    "productosVendidos": 180,
    "totalVentas": 10320
  },
  {
    "region": "CABA",
    "productosVendidos": 282,
    "totalVentas": 31572
  },
  {
    "region": "NEA",
    "productosVendidos": 420,
    "totalVentas": 56700
  }
]
```

Se decidió realizar el punto con la variable “totalVentas”, ya que el total de los productos vendidos eran órdenes de magnitud menor que 10000.

4. Se requiere obtener un reporte que contenga la siguiente información, nro. cuit, apellido y nombre y región y cantidad de facturas, ordenado por apellido.

```
db.facturas.aggregate([
  {
    $group: {
      _id: "$cliente.cuit",
      nombre: { $first: "$cliente.nombre" },
      apellido: { $first: "$cliente.apellido" },
      region: { $first: "$cliente.region" },
      cantidadFacturas: { $sum: 1 },
    },
  },
  { $sort: { apellido: 1 } },
  {
    $project: {
      _id: 0,
      cliente: { $concat: ["$apellido", " ", "$nombre"] },
      cuit: "$_id",
      region: "$region",
      cantidadFacturas: "$cantidadFacturas",
    },
  },
]);
```


```
[
  {
    "cliente": "Lavagno Soledad",
    "cuit": 2729887543,
    "region": "NOA",
    "cantidadFacturas": 14
  },
  {
    "cliente": "Malinez Marina",
    "cuit": 2740488484,
    "region": "CENTRO",
    "cantidadFacturas": 15
  },
  {
```

```
    "cliente": "Manoni Juan Manuel",
    "cuit": 2029889382,
    "region": "NEA",
    "cantidadFacturas": 42
  },
  {
    "cliente": "Zavasi Martin",
    "cuit": 2038373771,
    "region": "CABA",
    "cantidadFacturas": 29
  }
]
```

5. Basados en la consulta del punto 4 informar sólo los clientes con número de CUIT mayor a 27000000000.

```
db.facturas.aggregate([
  {
    $group: {
      _id: "$cliente.cuit",
      nombre: { $first: "$cliente.nombre" },
      apellido: { $first: "$cliente.apellido" },
      region: { $first: "$cliente.region" },
      cantidadFacturas: { $sum: 1 },
    },
  },
  { $match: { _id: { $gt: 2700000000 } } },
  { $sort: { apellido: 1 } },
  {
    $project: {
      _id: 0,
      cliente: { $concat: ["$apellido", " ", "$nombre"] },
      cuit: "$_id",
      region: "$region",
      cantidadFacturas: "$cantidadFacturas",
    },
  },
]);
```

```
[
  {
    "cliente": "Lavagno Soledad",
    "cuit": 2729887543,
    "region": "NOA",
    "cantidadFacturas": 14
  },
  {
    "cliente": "Malinez Marina",
    "cuit": 2740488484,
    "region": "CENTRO",
    "cantidadFacturas": 15
  }
]
```


Al realizar la consulta se detectó que no habia ningún cuit > 27000000000. por lo que se optó por quitar 1 cero para hacer la comparación del cuit.

6. Basados en la consulta del punto 5 informar solamente la cantidad de clientes que cumplen con esta condición.

```
db.facturas.aggregate([
  {
    $group: {
      _id: "$cliente.cuit",
      nombre: { $first: "$cliente.nombre" },
      apellido: { $first: "$cliente.apellido" },
      region: { $first: "$cliente.region" },
      cantidadFacturas: { $sum: 1 },
    },
  },
  { $match: { _id: { $gt: 2700000000 } } },
  { $count: "totalClientes" },
]);
```


```
[
  {
    "totalClientes": 2
  }
]
```

7. Se requiere realizar una consulta que devuelva la siguiente información: producto y cantidad de facturas en las que lo compraron, ordenado por cantidad de facturas descendente.

```
db.facturas.aggregate([
  { $unwind: "$item" },
  {
    $group: {
      _id: {
        producto: "$item.producto",
        factura: "$nroFactura",
      },
    },
  },
  {
    $group: {
      _id: "$_id.producto",
      cantidadFacturas: { $sum: 1 },
    },
  },
  { $sort: { cantidadFacturas: -1, _id: 1 } },
  {
    $project: {
      _id: 0,
      producto: "$_id",
      cantidadFacturas: "$cantidadFacturas",
    },
  },
]);
```

```
[
  {
    "producto": "TALADRO 12mm",
    "cantidadFacturas": 43
  },
  {
    "producto": "CORREA 10mm",
    "cantidadFacturas": 29
  },
  {
    "producto": "SET HERRAMIENTAS",
    "cantidadFacturas": 28
```

```
  },
  {
    "producto": "TUERCA 2mm",
    "cantidadFacturas": 28
  },
  {
    "producto": "TUERCA 5mm",
    "cantidadFacturas": 28
  },
  {
    "producto": " CORREA 12mm",
    "cantidadFacturas": 15
  }
]
```

8. Obtener la cantidad total comprada así como también los ingresos totales para cada producto.

```
db.facturas.aggregate([
  { $unwind: "$item" },
  {
    $group: {
      _id: "$item.producto",
      cantidad: { $sum: "$item.cantidad" },
      ingresoTotal: { $sum: { $multiply: ["$item.cantidad",
"$item.precio"] } },
    },
  },
  {
    $project: {
      _id: 0,
      producto: "$_id",
      cantidadTotal: "$cantidad",
      ingresoTotal: 1,
    },
  },
]);
```

```
[
  {
    "ingresoTotal": 19600,
    "producto": "SET HERRAMIENTAS",
    "cantidadTotal": 28
  },
  {
    "ingresoTotal": 21070,
    "producto": "TALADRO 12mm",
    "cantidadTotal": 43
  },
  {
    "ingresoTotal": 2970,
    "producto": " CORREA 12mm",
    "cantidadTotal": 165
  },
  {
    "ingresoTotal": 26532,
    "producto": "CORREA 10mm",
    "cantidadTotal": 198
  },
  {
    "ingresoTotal": 6720,
    "producto": "TUERCA 2mm",
    "cantidadTotal": 112
  },
  {
    "ingresoTotal": 31500,
    "producto": "TUERCA 5mm",
    "cantidadTotal": 350
  }
]
```

9. Idem el punto anterior, ordenar por ingresos en forma ascendente, saltear el 1ro y mostrar 2do y 3ro.

```
db.facturas.aggregate([
  { $unwind: "$item" },
  {
    $group: {
      _id: "$item.producto",
      cantidad: { $sum: "$item.cantidad" },
      ingresoTotal: { $sum: { $multiply: ["$item.cantidad",
"$item.precio"] } },
    },
  },
  { $sort: { ingresoTotal: 1 } },
  { $skip: 1 },
  { $limit: 2 },
  {
    $project: {
      _id: 0,
      producto: "$_id",
      cantidadTotal: "$cantidad",
      ingresoTotal: 1,
    },
  },
]);
```

```
[
  {
    "ingresoTotal": 6720,
    "producto": "TUERCA 2mm",
    "cantidadTotal": 112
  },
  {
    "ingresoTotal": 19600,
    "producto": "SET HERRAMIENTAS",
    "cantidadTotal": 28
  }
]
```

10. Obtener todos productos junto con un array de las personas que lo compraron. En este array deberá haber solo strings con el nombre completo de la persona. Los documentos entregados como resultado deberán tener la siguiente forma: {producto: “”, personas:[“…”, …]}

```
db.facturas.aggregate([
  { $unwind: "$item" },
  {
    $group: {
      _id: "$item.producto",
      personas: {
        $addToSet: { $concat: ["$cliente.nombre", " ",
"$cliente.apellido"] },
      },
    },
  },
  {
    $project: {
      _id: 0,
      producto: "$_id",
      personas: "$personas",
    },
  },
]);
```

```
[
  {
    "producto": "SET HERRAMIENTAS",
    "personas": [
      "Juan Manuel Manoni",
      "Soledad Lavagno"
    ]
  },
  {
    "producto": "TALADRO 12mm",
    "personas": [
      "Juan Manuel Manoni",
      "Marina Malinez"
    ]
  },
  {
```

```
    "producto": " CORREA 12mm",
    "personas": [
      "Marina Malinez"
    ]
  },
  {
    "producto": "CORREA 10mm",
    "personas": [
      "Martin Zavasi"
    ]
  },
  {
    "producto": "TUERCA 2mm",
    "personas": [
      "Juan Manuel Manoni",
      "Martin Zavasi"
    ]
  },
  {
    "producto": "TUERCA 5mm",
    "personas": [
      "Juan Manuel Manoni"
    ]
  }
]
```

11. Obtener los productos ordenados en forma descendente por la cantidad de diferentes personas que los compraron.

```
db.facturas.aggregate([
  { $unwind: "$item" },
  {
    $group: {
      _id: "$item.producto",
      personas: {
        $addToSet: { $concat: ["$cliente.nombre", " ",
"$cliente.apellido"] },
      },
    },
  },
  {
    $project: {
```

```
      _id: 0,
      producto: "$_id",
      personas: "$personas",
      cantidad: { $size: "$personas" },
    },
  },
  { $sort: { cantidad: -1 } },
]);
```


```
[
  {
    "producto": "TUERCA 2mm",
    "personas": [
      "Martin Zavasi",
      "Juan Manuel Manoni"
    ],
    "cantidad": 2
  },
  {
    "producto": "SET HERRAMIENTAS",
    "personas": [
      "Soledad Lavagno",
      "Juan Manuel Manoni"
    ],
    "cantidad": 2
  },
  {
    "producto": "TALADRO 12mm",
    "personas": [
      "Juan Manuel Manoni",
      "Marina Malinez"
    ],
    "cantidad": 2
  },
  {
    "producto": "CORREA 10mm",
    "personas": [
      "Martin Zavasi"
    ],
    "cantidad": 1
  },
  {
```

```
    "producto": " CORREA 12mm",
    "personas": [
      "Marina Malinez"
    ],
    "cantidad": 1
  },
  {
    "producto": "TUERCA 5mm",
    "personas": [
      "Juan Manuel Manoni"
    ],
    "cantidad": 1
  }
]
```

12. Obtener el total gastado por persona y mostrar solo los que gastaron más de 3100000. Los documentos devueltos deben tener el nombre completo del cliente y el total gastado: {cliente:"",total:}

```
db.facturas.aggregate([
  { $unwind: "$item" },
  {
    $group: {
      _id: "$cliente.cuit",
      cliente: {
        $first: { $concat: ["$cliente.nombre", " ",
"$cliente.apellido"] },
      },
      gasto: { $sum: { $multiply: ["$item.cantidad",
"$item.precio"] } },
    },
  },
  { $match: { gasto: { $gt: 31000 } } },
  {
    $project: {
      _id: 0,
      cliente: "$cliente",
      total: "$gasto",
    },
  },
]);
```

```
[
  {
    "cliente": "Martin Zavasi",
    "total": 31572
  },
  {
    "cliente": "Juan Manuel Manoni",
    "total": 56700
  }
]
```

13. Obtener el promedio de gasto por factura por cada región.

```
db.facturas.aggregate([
  { $unwind: "$item" },
  {
    $group: {
      _id: "$nroFactura",
      region: { $first: "$cliente.region" },
      precioFactura: {
        $sum: { $multiply: ["$item.cantidad", "$item.precio"] },
      },
    },
  },
  {
    $group: {
      _id: "$region",
      promedioFactura: {
        $avg: "$precioFactura",
      },
    },
  },
  {
    $project: {
      _id: 0,
      region: "$_id",
      promedioFactura: { $round: ["$promedioFactura", 2] },
    },
  },
]);
```

```
[
  {
    "region": "NEA",
    "promedioFactura": 1350
  },
  {
    "region": "CENTRO",
    "promedioFactura": 688
  },
  {
    "region": "CABA",
    "promedioFactura": 1088.69
  },
  {
    "region": "NOA",
    "promedioFactura": 700
  }
]
```


14. Obtener la factura en la que se haya gastado más. En caso de que sean varias obtener la que tenga el número de factura menor.

```
db.facturas.aggregate([
  { $unwind: "$item" },
  {
    $group: {
      _id: "$nroFactura",
      gastoTotalFactura: {
        $sum: { $multiply: ["$item.cantidad", "$item.precio"] },
      },
    },
  },
  { $sort: { gastoTotalFactura: -1, _id: 1 } },
  { $limit: 1 },
  {
    $project: {
      _id: 0,
      nroFactura: "$_id",
      gastoTotalFactura: "$gastoTotalFactura",
    },
  },
]);
```

```
[
  {
    "nroFactura": 1002,
    "gastoTotalFactura": 1968
  }
]
```

15. Obtener a los clientes indicando cuánto fue lo que más gastó en una única factura.

```
db.facturas.aggregate([
  { $unwind: "$item" },
  {
    $group: {
      _id: "$nroFactura",
      cliente: {
        $first: { $concat: ["$cliente.nombre", " ",
"$cliente.apellido"] },
      },
      gastoTotalFactura: {
        $sum: { $multiply: ["$item.cantidad", "$item.precio"] },
      },
    },
  },
  {
    $group: {
      _id: "$cliente",
      mayorGastoFactura: { $max: "$gastoTotalFactura" },
    },
  },
  { $sort: { mayorGastoFactura: -1 } },
  {
    $project: {
      _id: 0,
      cliente: "$_id",
      mayorGastoFactura: "$mayorGastoFactura",
    },
  },
]);
```

```
[
  {
    "cliente": "Martin Zavasi",
    "mayorGastoFactura": 1968
  },
  {
    "cliente": "Juan Manuel Manoni",
    "mayorGastoFactura": 1960
  },
  {
    "cliente": "Soledad Lavagno",
    "mayorGastoFactura": 700
  },
  {
    "cliente": "Marina Malinez",
    "mayorGastoFactura": 688
  }
]
```

16. Utilizando MapReduce, indicar la cantidad total comprada de cada ítem. Comparar el resultado con el ejercicio 8.

```
db.facturas.mapReduce(
  function () {
    this.item.forEach((producto) => {
      emit(producto.producto, producto.cantidad);
    });
  },
  function (key, valor) {
    return Array.sum(valor);
  },
  { out: "totalProductosVendidos" }
);

db.totalProductosVendidos.find();
```

17. Obtener la información de los clientes que hayan gastado 100000 en una orden junto con el número de orden.

    Se modifico el valor total de la factura de 10000 a 1000.

```
db.facturas.aggregate([
```

```
  { $unwind: "$item" },
  {
    $group: {
      _id: "$nroFactura",
      cliente: {
        $first: { $concat: ["$cliente.nombre", " ",
"$cliente.apellido"] },
      },
      region: { $first: "$cliente.region" },
      cuit: { $first: "$cliente.cuit" },
      total: { $sum: { $multiply: ["$item.precio",
"$item.cantidad"] } },
    },
  },
  { $match: { total: { $gt: 1000 } } },
  {
    $project: {
      _id: 0,
      cliente: "$cliente",
      cuit: "$cuit",
      region: "$region",
      nroFactura: "$_id",
      total: "$total",
    },
  },
]);
```

```
[
  {
    "cliente": "Juan Manuel Manoni",
    "cuit": 2029889382,
    "region": "NEA",
    "nroFactura": 1010,
    "total": 1960
  },
  {
    "cliente": "Juan Manuel Manoni",
    "cuit": 2029889382,
    "region": "NEA",
    "nroFactura": 1031,
    "total": 1960
  },
```

```
{
  "cliente": "Juan Manuel Manoni",
  "cuit": 2029889382,
  "region": "NEA",
  "nroFactura": 1012,
  "total": 1190
},
{
  "cliente": "Juan Manuel Manoni",
  "cuit": 2029889382,
  "region": "NEA",
  "nroFactura": 1066,
  "total": 1960
},
{
  "cliente": "Martin Zavasi",
  "cuit": 2038373771,
  "region": "CABA",
  "nroFactura": 1086,
  "total": 1968
},
{
  "cliente": "Juan Manuel Manoni",
  "cuit": 2029889382,
  "region": "NEA",
  "nroFactura": 1087,
  "total": 1960
},
{
  "cliente": "Juan Manuel Manoni",
  "cuit": 2029889382,
  "region": "NEA",
  "nroFactura": 1089,
  "total": 1190
},
{
  "cliente": "Martin Zavasi",
  "cuit": 2038373771,
  "region": "CABA",
  "nroFactura": 1093,
  "total": 1968
},
```

```
{
  "cliente": "Juan Manuel Manoni",
  "cuit": 2029889382,
  "region": "NEA",
  "nroFactura": 1038,
  "total": 1960
},
{
  "cliente": "Martin Zavasi",
  "cuit": 2038373771,
  "region": "CABA",
  "nroFactura": 1044,
  "total": 1968
},
{
  "cliente": "Juan Manuel Manoni",
  "cuit": 2029889382,
  "region": "NEA",
  "nroFactura": 1068,
  "total": 1190
},
{
  "cliente": "Martin Zavasi",
  "cuit": 2038373771,
  "region": "CABA",
  "nroFactura": 1016,
  "total": 1968
},
{
  "cliente": "Juan Manuel Manoni",
  "cuit": 2029889382,
  "region": "NEA",
  "nroFactura": 1045,
  "total": 1960
},
{
  "cliente": "Juan Manuel Manoni",
  "cuit": 2029889382,
  "region": "NEA",
  "nroFactura": 1052,
  "total": 1960
},
```

```
{
  "cliente": "Juan Manuel Manoni",
  "cuit": 2029889382,
  "region": "NEA",
  "nroFactura": 1096,
  "total": 1190
},
{
  "cliente": "Martin Zavasi",
  "cuit": 2038373771,
  "region": "CABA",
  "nroFactura": 1058,
  "total": 1968
},
{
  "cliente": "Juan Manuel Manoni",
  "cuit": 2029889382,
  "region": "NEA",
  "nroFactura": 1094,
  "total": 1960
},
{
  "cliente": "Juan Manuel Manoni",
  "cuit": 2029889382,
  "region": "NEA",
  "nroFactura": 1033,
  "total": 1190
},
{
  "cliente": "Martin Zavasi",
  "cuit": 2038373771,
  "region": "CABA",
  "nroFactura": 1065,
  "total": 1968
},
{
  "cliente": "Martin Zavasi",
  "cuit": 2038373771,
  "region": "CABA",
  "nroFactura": 1002,
  "total": 1968
}
```

```
]
```

18. En base a la localidad de los clientes, obtener el total facturado por localidad.

```
db.facturas.aggregate([
  { $unwind: "$item" },
  {
    $group: {
      _id: "$cliente.region",
      region: { $first: "$cliente.region" },
      total: { $sum: { $multiply: ["$item.precio",
"$item.cantidad"] } },
    },
  },
  {
    $project: {
      _id: 0,
      region: "$_id",
      total: "$total",
    },
  },
]);
```

```
[
  {
    "region": "NEA",
    "total": 56700
  },
  {
    "region": "CENTRO",
    "total": 10320
  },
  {
    "region": "NOA",
    "total": 9800
  },
  {
    "region": "CABA",
    "total": 31572
  }
]
```