



---

CARRERA DE ESPECIALIZACIÓN EN INTELIGENCIA ARTIFICIAL

---

## **Bases de Datos para Inteligencia Artificial**

16Co2024

### **Trabajo Práctico N°2**

#### ***MongoDB***

DEGANO, MYRNA LORENA

N° SIU a1618

[myrna.l.degano@gmail.com](mailto:myrna.l.degano@gmail.com)

abril de 2025

## Índice

SetUp.....	2
Consultas.....	3
1) .....	3
2) .....	4
3) .....	5
4) .....	6
5) .....	8
6) .....	10
7) .....	11
8) .....	12
9) .....	13
10) .....	14
11).....	15
12) .....	16
13) .....	18
14) .....	19
15) .....	20
16) (Opcional) .....	21
17) (Opcional) .....	23
18) (Opcional) .....	25
Anexos.....	26
Referencias.....	26

## SetUp

Para crear el entorno de trabajo con Docker, ejecutar:

```
docker-compose up -d
```

Esto crea el contenedor con **MongoDB** inicializado con la base de datos **finanzas** y la colección **facturas** con los datos provistos en el archivo *json*.

```
myrna@myrna-Latitude-E5470:~/Projects/BDIA/BDIA-TP2$ docker ps -a
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS
a9b17020ce6b   mongo:latest   "docker-entrypoint.s..." 2 minutes ago  Up 2 minutes (healthy)  0.0.0.0:27017->27017/tcp
p, [::]:27017->27017/tcp   mongo_container
```

Para acceder al contenedor con Mongo Shell:

```
docker exec -it mongo_container mongosh -u mongo -p mongo --
authenticationDatabase admin
```

Acceder a la base de datos creada para el trabajo práctico:

```
use finanzas
```

```
test> use finanzas
switched to db finanzas
finanzas> █
```

## Consultas

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",
      total: {
        $sum: "$item.cantidad"
      }
    }
  },
  {
    $project: {
      region: "$_id",
      total: 1,
      _id: 0
    }
  }
])
```

```
finanzas> db.facturas.aggregate([
...   {
...     $unwind: "$item"
...   },
...   {
...     $group: {
...       _id: "$cliente.region",
...       total: {
...         $sum: "$item.cantidad"
...       }
...     }
...   },
...   {
...     $project: {
...       region: "$_id",
...       total: 1,
...       _id: 0
...     }
...   }
... ])
[
  { total: 180, region: 'CENTRO' },
  { total: 282, region: 'CABA' },
  { total: 420, region: 'NEA' },
  { total: 14, region: 'NOA' }
]
finanzas> 
```

DL-YOUTUBE

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",
      total: {
        $sum: "$item.cantidad"
      }
    }
  },
  {
    $project: {
      region: "$_id",
      total: 1,
      _id: 0
    }
  },
  {
    $sort: {
      total: 1
    }
  },
  {
    $limit: 1
  }
])
```

```
finanzas> db.facturas.aggregate([
...   {
...     $unwind: "$item"
...   },
...   {
...     $group: {
...       _id: "$cliente.region",
...       total: {
...         $sum: "$item.cantidad"
...       }
...     }
...   },
...   {
...     $project: {
...       region: "$_id",
...       total: 1,
...       _id: 0
...     }
...   },
...   {
...     $sort: {
...       total: 1
...     }
...   },
...   {
...     $limit: 1
...   }
... ])
[ { total: 14, region: 'NOA' } ]
```

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",
      total: {
        $sum: "$item.cantidad"
      }
    }
  },
  {
    $project: {
      region: "$_id",
      total: 1,
      _id: 0
    }
  },
  {
    $match: {
      total: {
        $gt: 10000
      }
    }
  }
])
```

\* *No hay resultados* – Alternativa: Cantidad superior a 300

```
db.facturas.aggregate([
  {
    $unwind: "$item"
  },
  {
    $group: {
      _id: "$cliente.region",
      total: {
        $sum: "$item.cantidad"
      }
    }
  },
  {
    $project: {
      region: "$_id",
      total: 1,
      _id: 0
    }
  },
  {
    $match: {
```

```

        total: {
            $gt: 300
        }
    }
}
] )

```

```

finanzas> db.facturas.aggregate([
...   {
...     $unwind: "$item"
...   },
...   {
...     $group: {
...       _id: "$cliente.region",
...       total: {
...         $sum: "$item.cantidad"
...       }
...     }
...   },
...   {
...     $project: {
...       region: "$_id",
...       total: 1,
...       _id: 0
...     }
...   },
...   {
...     $match: {
...       total: {
...         $gt: 300
...       }
...     }
...   }
... ])
[ { total: 420, region: 'NEA' } ]

```

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([
  {
    $group: {
      _id: {
        apellido: "$cliente.apellido",
        nombre: "$cliente.nombre",
        region: "$cliente.region",
        cuit: "$cliente.cuit"
      },
      cantFacturas: {
        $sum: 1
      }
    }
  },
  {
    $project: {
      apellido: "$_id.apellido",

```

```
    nombre: "$_id.nombre",
    region: "$_id.region",
    cuit: "$_id.cuit",
    cantFacturas: 1,
    _id: 0
  }
},
{
  $sort: {
    apellido: 1
  }
}
])
```

```
[
  {
    cantFacturas: 14,
    apellido: 'Lavagno',
    nombre: 'Soledad',
    region: 'NOA',
    cuit: 2729887543
  },
  {
    cantFacturas: 15,
    apellido: 'Malínez',
    nombre: 'Marina',
    region: 'CENTRO',
    cuit: 2740488484
  },
  {
    cantFacturas: 42,
    apellido: 'Manoni',
    nombre: 'Juan Manuel',
    region: 'NEA',
    cuit: 2029889382
  },
  {
    cantFacturas: 29,
    apellido: 'Zavasi',
    nombre: 'Martin',
    region: 'CABA',
    cuit: 2038373771
  }
]
finanzas> □
```



5)

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

```
db.facturas.aggregate([
  {
    $group: {
      _id: {
        apellido: "$cliente.apellido",
        nombre: "$cliente.nombre",
        region: "$cliente.region",
        cuit: "$cliente.cuit"
      },
      cantFacturas: {
        $sum: 1
      }
    }
  },
  {
    $project: {
      apellido: "$_id.apellido",
      nombre: "$_id.nombre",
      region: "$_id.region",
      cuit: "$_id.cuit",
      cantFacturas: 1,
      _id: 0
    }
  },
  {
    $match: {
      cuit: {
        $gt: 27000000000
      }
    }
  },
  {
    $sort: {
      apellido: 1
    }
  }
])
```

\* **No hay resultados** – Alternativa: CUIT mayor a 27000000000

```
db.facturas.aggregate([
  {
    $group: {
      _id: {
        apellido: "$cliente.apellido",
        nombre: "$cliente.nombre",
        region: "$cliente.region",
        cuit: "$cliente.cuit"
      },
    },
```

```
    cantFacturas: {
      $sum: 1
    }
  },
  {
    $project: {
      apellido: "$_id.apellido",
      nombre: "$_id.nombre",
      region: "$_id.region",
      cuit: "$_id.cuit",
      cantFacturas: 1,
      _id: 0
    }
  },
  {
    $match: {
      cuit: {
        $gt: 27000000000
      }
    }
  },
  {
    $sort: {
      apellido: 1
    }
  }
}
])
```

```
[
  {
    cantFacturas: 14,
    apellido: 'Lavagno',
    nombre: 'Soledad',
    region: 'NOA',
    cuit: 2729887543
  },
  {
    cantFacturas: 15,
    apellido: 'Malínez',
    nombre: 'Marina',
    region: 'CENTRO',
    cuit: 2740488484
  }
]
finanzas> □
```

6)

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

```
db.facturas.aggregate([
  {
    $group: {
      _id: {
        apellido: "$cliente.apellido",
        nombre: "$cliente.nombre",
        region: "$cliente.region",
        cuit: "$cliente.cuit"
      },
      cantFacturas: {
        $sum: 1
      }
    }
  },
  {
    $project: {
      apellido: "$_id.apellido",
      nombre: "$_id.nombre",
      region: "$_id.region",
      cuit: "$_id.cuit",
      cantFacturas: 1,
      _id: 0
    }
  },
  {
    $match: {
      cuit: {
        $gt: 27000000000
      }
    }
  },
  {
    $count: "totalClientes"
  }
])
```

```
...     cantFacturas: 1,
...     _id: 0
...   }
... },
... {
...   $match: {
...     cuit: {
...       $gt: 27000000000
...     }
...   }
... },
... {
...   $count: "totalClientes"
... }
... ])
[ { totalClientes: 2 } ]
finanzas> □
```

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",
      cantFacturas: {
        $sum: 1
      }
    }
  },
  {
    $project: {
      producto: "$_id",
      cantFacturas: 1,
      _id: 0
    }
  },
  {
    $sort: {
      cantFacturas: -1
    }
  }
])
```

```
[
  { cantFacturas: 43, producto: 'TALADRO 12mm' },
  { cantFacturas: 29, producto: 'CORREA 10mm' },
  { cantFacturas: 28, producto: 'TUERCA 2mm' },
  { cantFacturas: 28, producto: 'TUERCA 5mm' },
  { cantFacturas: 28, producto: 'SET HERRAMIENTAS' },
  { cantFacturas: 15, producto: 'CORREA 12mm' }
]
finanzas> 
```

8)

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

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

```
[
  { totalCantidad: 350, totalIngresos: 31500, producto: 'TUERCA 5mm' },
  { totalCantidad: 198, totalIngresos: 26532, producto: 'CORREA 10mm' },
  { totalCantidad: 43, totalIngresos: 21070, producto: 'TALADRO 12mm' },
  {
    totalCantidad: 28,
    totalIngresos: 19600,
    producto: 'SET HERRAMIENTAS'
  },
  { totalCantidad: 112, totalIngresos: 6720, producto: 'TUERCA 2mm' },
  { totalCantidad: 165, totalIngresos: 2970, producto: 'CORREA 12mm' }
]
finanzas> □
```

9)

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

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

```
[
  { totalCantidad: 112, totalIngresos: 6720, producto: 'TUERCA 2mm' },
  {
    totalCantidad: 28,
    totalIngresos: 19600,
    producto: 'SET HERRAMIENTAS'
  }
]
finanzas> □
```

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: "<nombre>", personas: ["...", ...]}
```

```
db.facturas.aggregate([
  {
    $unwind: "$item"
  },
  {
    $group: {
      _id: "$item.producto",
      personas: {
        $addToSet: {
          $concat: [
            "$cliente.nombre", " ", "$cliente.apellido"
          ]
        }
      }
    }
  },
  {
    $project: {
      producto: "$_id",
      personas: 1,
      _id: 0
    }
  }
])
```

```
... {
...   $project: {
...     producto: "$_id",
...     personas: 1,
...     _id: 0
...   }
... }
... ])
[
  { personas: [ 'Marina Malinez' ], producto: 'CORREA 12mm' },
  { personas: [ 'Juan Manuel Manoni' ], producto: 'TUERCA 5mm' },
  {
    personas: [ 'Juan Manuel Manoni', 'Soledad Lavagno' ],
    producto: 'SET HERRAMIENTAS'
  },
  { personas: [ 'Martín Zavasi' ], producto: 'CORREA 10mm' },
  {
    personas: [ 'Juan Manuel Manoni', 'Martín Zavasi' ],
    producto: 'TUERCA 2mm'
  },
  {
    personas: [ 'Marina Malinez', 'Juan Manuel Manoni' ],
    producto: 'TALADRO 12mm'
  }
]
finanzas> □
```

11)

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

```
db.facturas.aggregate([
  {
    $unwind: "$item"
  },
  {
    $group: {
      _id: {
        producto: "$item.producto",
        persona: {
          $concat: ["$cliente.nombre", " ", "$cliente.apellido"]
        }
      }
    }
  },
  {
    $group: {
      _id: "$_id.producto",
      cantidadPersonas: { $sum: 1 }
    }
  },
  {
    $project: {
      producto: "$_id",
      cantidadPersonas: 1,
      _id: 0
    }
  },
  {
    $sort: {
      cantidadPersonas: -1
    }
  }
])
```

```
[
  { cantidadPersonas: 2, producto: 'TUERCA 2mm' },
  { cantidadPersonas: 2, producto: 'SET HERRAMIENTAS' },
  { cantidadPersonas: 2, producto: 'TALADRO 12mm' },
  { cantidadPersonas: 1, producto: 'CORREA 10mm' },
  { cantidadPersonas: 1, producto: 'CORREA 12mm' },
  { cantidadPersonas: 1, producto: 'TUERCA 5mm' }
]
finanzas> 
```



## 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:"<nombreCompleto>",total:<num>}

```
db.facturas.aggregate([
  {
    $unwind: "$item"
  },
  {
    $group: {
      _id: {
        nombreCompleto: {
          $concat: ["$cliente.nombre", " ", "$cliente.apellido"]
        }
      },
      totalGastado: {
        $sum: { $multiply: ["$item.cantidad", "$item.precio"] }
      }
    }
  },
  {
    $match: {
      totalGastado: {
        $gt: 3100000
      }
    }
  },
  {
    $project: {
      cliente: "$_id.nombreCompleto",
      total: "$totalGastado",
      _id: 0
    }
  }
])
```

\* *No hay resultados* – Alternativa: Gastaron más de 31000

```
db.facturas.aggregate([
  {
    $unwind: "$item"
  },
  {
    $group: {
      _id: {
        nombreCompleto: {
          $concat: ["$cliente.nombre", " ", "$cliente.apellido"]
        }
      },
      totalGastado: {
        $sum: { $multiply: ["$item.cantidad", "$item.precio"] }
      }
    }
  }
])
```

```
}
},
{
  $match: {
    totalGastado: {
      $gt: 31000
    }
  }
},
{
  $project: {
    cliente: "$_id.nombreCompleto",
    total: "$totalGastado",
    _id: 0
  }
}
])
```

```
... {
...   $match: {
...     totalGastado: {
...       $gt: 31000
...     }
...   },
...   {
...     $project: {
...       cliente: "$_id.nombreCompleto",
...       total: "$totalGastado",
...       _id: 0
...     }
...   }
... }
... ])
[
  { cliente: 'Martin Zavasi', total: 31572 },
  { cliente: 'Juan Manuel Manoni', total: 56700 }
]
finanzas> 
```

13)

Obtener el promedio de gasto por factura por cada región.

```

db.facturas.aggregate([
  {
    $unwind: "$item"
  },
  {
    $group: {
      _id: "$_id",
      region: {
        $first: "$cliente.region"
      },
      totalFactura: {
        $sum: {
          $multiply: ["$item.cantidad", "$item.precio"]
        }
      }
    }
  },
  {
    $group: {
      _id: "$region",
      totalGastado: {
        $sum: "$totalFactura"
      },
      cantidadFacturas: {
        $sum: 1
      }
    }
  },
  {
    $project: {
      region: "$_id",
      promedioPorFactura: {
        $divide: ["$totalGastado", "$cantidadFacturas"]
      },
      _id: 0
    }
  }
])

```

```

...   }
...   },
...   {
...     $project: {
...       region: "$_id",
...       promedioPorFactura: {
...         $divide: ["$totalGastado", "$cantidadFacturas"]
...       },
...       _id: 0
...     }
...   }
... })
[
  { region: 'CENTRO', promedioPorFactura: 688 },
  { region: 'CABA', promedioPorFactura: 1088.6896551724137 },
  { region: 'NEA', promedioPorFactura: 1350 },
  { region: 'NOA', promedioPorFactura: 700 }
]
finanzas> 

```

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"
  },
  {
    $group: {
      _id: "$_id",
      nroFactura: {
        $first: "$nroFactura"
      },
      cliente: {
        $first: "$cliente"
      },
      totalFactura: {
        $sum: {
          $multiply: ["$item.cantidad", "$item.precio"]
        }
      }
    }
  },
  {
    $sort: {
      totalFactura: -1,
      nroFactura: 1
    }
  },
  {
    $limit: 1
  },
  {
    $project: {
      _id: 0,
      nroFactura: 1,
      cliente: {
        nombreCompleto: {
          $concat: ["$cliente.nombre", " ", "$cliente.apellido"]
        },
        region: "$cliente.region"
      },
      totalFactura: 1
    }
  }
])
```

```

...   _id: 0,
...   nroFactura: 1,
...   cliente: {
...     nombreCompleto: {
...       $concat: ["$cliente.nombre", " ", "$cliente.apellido"]
...     },
...     region: "$cliente.region"
...   },
...   totalFactura: 1
... }
... }
... })
[
  {
    nroFactura: 1002,
    cliente: { nombreCompleto: 'Martin Zavasi', region: 'CABA' },
    totalFactura: 1968
  }
]
finanzas> 

```

15)

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

```

db.facturas.aggregate([
  {
    $unwind: "$item"
  },
  {
    $group: {
      _id: {
        clienteId: "$cliente.cuit",
        nombreCompleto: {
          $concat: ["$cliente.nombre", " ", "$cliente.apellido"]
        },
        facturaId: "$_id"
      },
      totalFactura: {
        $sum: {
          $multiply: ["$item.cantidad", "$item.precio"]
        }
      }
    }
  },
  {
    $group: {
      _id: {
        clienteId: "$_id.clienteId",
        nombreCompleto: "$_id.nombreCompleto"
      },
      gastoMaximo: {
        $max: "$totalFactura"
      }
    }
  },
  {
    $project: {
      _id: 0,

```

```

    cliente: "$_id.nombreCompleto",
    gastoMaximo: 1
  }
}
])

```

```

...     gastoMaximo: {
...         $max: "$totalFactura"
...     }
... },
... {
...     $project: {
...         _id: 0,
...         cliente: "$_id.nombreCompleto",
...         gastoMaximo: 1
...     }
... }
... ])
[
  { gastoMaximo: 1960, cliente: 'Juan Manuel Manoni' },
  { gastoMaximo: 688, cliente: 'Marina Malinez' },
  { gastoMaximo: 1968, cliente: 'Martin Zavasi' },
  { gastoMaximo: 700, cliente: 'Soledad Lavagno' }
]
finanzas> 

```

## 16) (Opcional)

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

```

var mapFunction = function () {
  this.item.forEach(function (producto) {
    emit(producto.producto, producto.cantidad);
  });
};

var reduceFunction = function (key, values) {
  return Array.sum(values);
};

db.facturas.mapReduce(
  mapFunction,
  reduceFunction,
  {
    out: "total_por_producto"
  }
);

db.total_por_producto.find().sort({ value: -1 })

```

```
... };
...

finanzas> var reduceFunction = function (key, values) {
...   return Array.sum(values);
... };
...

finanzas> db.facturas.mapReduce(
...   mapFunction,
...   reduceFunction,
...   {
...     out: "total_por_producto"
...   }
... );
...
DeprecationWarning: Collection.mapReduce() is deprecated. Use an aggregation instead.
See https://docs.mongodb.com/manual/core/map-reduce for details.
{ result: 'total_por_producto', ok: 1 }
finanzas> 
```

```
...   reduceFunction,
...   {
...     out: "total_por_producto"
...   }
... );
...
DeprecationWarning: Collection.mapReduce() is deprecated. Use an aggregation instead.
See https://docs.mongodb.com/manual/core/map-reduce for details.
{ result: 'total_por_producto', ok: 1 }
finanzas> db.total_por_producto.find().sort({ value: -1 })
...
[
  { _id: 'TUERCA 5mm', value: 350 },
  { _id: 'CORREA 10mm', value: 198 },
  { _id: 'CORREA 12mm', value: 165 },
  { _id: 'TUERCA 2mm', value: 112 },
  { _id: 'TALADRO 12mm', value: 43 },
  { _id: 'SET HERRAMIENTAS', value: 28 }
]
finanzas> 
```

\* Map-reduce está deprecado a partir de la versión 5.0.

<https://docs.mongodb.com/manual/core/map-reduce>

Alternativa: Aggregation Pipeline

```
db.facturas.aggregate([
  {
    $unwind: "$item"
  },
  {
    $group: {
      _id: "$item.producto",
      cantidadTotal: {
        $sum: "$item.cantidad"
      }
    }
  },
  {
    $project: {
```

```

    producto: "$_id",
    cantidadTotal: 1,
    _id: 0
  },
  {
    $sort: {
      cantidadTotal: -1
    }
  }
]
})

```

```

...     producto: "$_id",
...     cantidadTotal: 1,
...     _id: 0
...   }
... },
... {
...   $sort: {
...     cantidadTotal: -1
...   }
... }
... ])
[
  { cantidadTotal: 350, producto: 'TUERCA 5mm' },
  { cantidadTotal: 198, producto: 'CORREA 10mm' },
  { cantidadTotal: 165, producto: 'CORREA 12mm' },
  { cantidadTotal: 112, producto: 'TUERCA 2mm' },
  { cantidadTotal: 43, producto: 'TALADRO 12mm' },
  { cantidadTotal: 28, producto: 'SET HERRAMIENTAS' }
]
finanzas>

```

## 17) (Opcional)

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

*\* No hay resultados.* Alternativa: Asumo que “Orden” es lo mismo que “Factura” y que el total gastado sea igual a 1960.

```

db.facturas.aggregate([
  {
    $unwind: "$item"
  },
  {
    $group: {
      _id: "$_id",
      nroFactura: {
        $first: "$nroFactura"
      },
      cliente: {
        $first: {
          $concat: ["$cliente.nombre", " ", "$cliente.apellido"]
        }
      }
    }
  }
])

```



```
    },
    totalGastado: {
      $sum: {
        $multiply: ["$item.cantidad", "$item.precio"]
      }
    }
  }
},
{
  $match: {
    totalGastado: 1960
  }
},
{
  $project: {
    _id: 0,
    cliente: 1,
    nroFactura: 1,
    totalGastado: 1
  }
}
])
```

```
mongosh mongodb://<credentials>@127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=20000
cliente: 'Juan Manuel Manoni',
totalGastado: 1960
},
{
  nroFactura: 1017,
  cliente: 'Juan Manuel Manoni',
  totalGastado: 1960
},
{
  nroFactura: 1094,
  cliente: 'Juan Manuel Manoni',
  totalGastado: 1960
},
{
  nroFactura: 1066,
  cliente: 'Juan Manuel Manoni',
  totalGastado: 1960
},
{
  nroFactura: 1073,
  cliente: 'Juan Manuel Manoni',
  totalGastado: 1960
},
{
  nroFactura: 1087,
  cliente: 'Juan Manuel Manoni',
  totalGastado: 1960
},
{
  nroFactura: 1010,
  cliente: 'Juan Manuel Manoni',
  totalGastado: 1960
}
]
finanzas> 
```

## 18) (Opcional)



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

\* *No hay resultados.* Alternativa: Asumo que “Localidad” es “Región”.

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

```
... 1
...   $project: {
...     _id: 0,
...     region: "$_id",
...     totalFacturado: 1
...   }
... },
... {
...   $sort: {
...     totalFacturado: -1
...   }
... }
... })
[
  { totalFacturado: 56700, region: 'NEA' },
  { totalFacturado: 31572, region: 'CABA' },
  { totalFacturado: 10320, region: 'CENTRO' },
  { totalFacturado: 9800, region: 'NDA' }
]
finanzas> □
```

## Anexos

	<ul style="list-style-type: none"><li>• <a href="#">Archivo JS</a> con las consultas.</li></ul>
	<ul style="list-style-type: none"><li>• <a href="#">Repositorio GitHub</a> con el archivo de configuración del entorno y los inputs.</li></ul>

## Referencias

- [https://github.com/FIUBA-Posgrado-Inteligencia-Artificial/BDIA/tree/main/clase\\_5](https://github.com/FIUBA-Posgrado-Inteligencia-Artificial/BDIA/tree/main/clase_5)
- <https://www.mongodb.com/>