



Unidad Nacional Experimental de Guayana.  
Vicerrectorado Académico.  
Coordinación General de Pregrado.  
Proyecto de carrera: Ingeniería en Informática.  
Asignatura: Sistemas de bases de datos II – Sección: 1.

# Bases de Datos Documentales con MongoDB

Profesor:  
Clinia Cordero

Integrantes:  
V-25.933.680 Alburquerque Sheen  
V-31.818.222 Valencia Haddan  
V-31.445.710 Longart Daniela

Ciudad Guayana, 05 Diciembre del 2025

## **Introducción**

GlobalMarket, es una startup de comercio electrónico en expansión. Su base de datos relacional actual está fallando bajo la carga de consultas complejas y la variabilidad de los atributos de sus productos. Nuestra función es migrar y optimizar el catálogo de productos y el registro de ventas a MongoDB Atlas, implementando un motor de búsqueda eficiente y un dashboard de análisis de ventas en tiempo real.

Este proyecto consiste en la creación, gestión y consulta de una base de datos NoSQL utilizando MongoDB Atlas y el shell de comandos. Se trabajará con colecciones, documentos, consultas simples y avanzadas, operadores lógicos y actualizaciones, con el objetivo de comprender la estructura flexible de datos de MongoDB.

# Índice

Introducción .....	2
Índice .....	3
Objetivos Específicos .....	5
Descripción del problema .....	6
Requerimientos del sistema .....	7
Software .....	7
Hardware .....	7
Modelado y Validación .....	8
Desarrollo .....	10
b. DATOS DE PRUEBA PARA VALIDACIÓN .....	19
Documento INVÁLIDO (debe fallar): .....	19
c. Documento VÁLIDO (debe pasar): .....	20
8. Pipeline 3 – Bucket por rangos de precios unitarios .....	29
ÍNDICE 1: Compuesto (categoría + fecha) .....	32
ÍNDICE 2: Simple (product_id) .....	33
ÍNDICE 3: Compuesto (sales + quantity) .....	34
Observación .....	35
Explain plan .....	39
Paso 1: Cargar el pipeline .....	39
Paso 2: Ejecutar Explain .....	39
Estadísticas de ejecución .....	39
Plan rechazado. Muestra que MongoDB consideró otro índice (idx_category_date) pero eligió idx_sales_quantity porque estimó que era más eficiente.....	40
Visualización .....	42
1. Ventas por país (gráfico de barras – arriba izquierda) .....	42
2. Cantidad por ganancias (gráfico de rosquilla – arriba centro) .....	42
3. Productos vendidos (gráfico horizontal – arriba derecha) .....	43
4. Ventas Totales (parte inferior izquierda).....	43
Valor total de ventas: 12,642,501.9099 .....	43

5. Productos por fecha — Productos más pedidos por año (abajo centro).....	43
6. Producto por país – Mapa mundial (abajo derecha) .....	44
Recomendaciones generales.....	45
Conclusión.....	46

# **Objetivos**

## **Objetivo General**

Crear y gestionar una base de datos NoSQL mediante comandos en MongoDB Shell utilizando MongoDB Atlas como servidor remoto.

## **Objetivos Específicos**

- Conectar un cliente local a una base de datos en la nube.
- Crear una base de datos y colecciones.
- Insertar documentos utilizando formato JSON.
- Aplicar consultas simples y avanzadas con operadores.
- Actualizar, eliminar y filtrar documentos.
- Evaluar el funcionamiento del CRUD completo.

## Descripción del problema

GlobalMarket, una empresa dedicada a la comercialización de productos de diferentes categorías, gestiona información clave relacionada con inventarios, precios, disponibilidad de productos y características comerciales. Hasta el momento, estos datos han sido almacenados en un sistema tradicional que presenta limitaciones al momento de realizar consultas avanzadas, aplicar cambios estructurales o integrar nueva información de forma dinámica.

Debido al crecimiento de la empresa y a la necesidad de realizar análisis más completos y deducciones automáticas sobre los datos almacenados, se plantea una migración de la base de datos actual hacia un sistema NoSQL basado en MongoDB.

MongoDB permite trabajar con una base de datos deductiva, capaz de almacenar y procesar datos de forma flexible mediante documentos no estructurados, lo cual facilita la interpretación de la información, el establecimiento de relaciones implícitas y la adaptación a nuevos requerimientos sin afectar la continuidad del sistema. Asimismo, su implementación en la nube mediante MongoDB Atlas garantiza accesibilidad remota, disponibilidad continua y una gestión centralizada más eficiente.

La falta de una solución moderna que soporte consultas más inteligentes y flexibles limita actualmente el aprovechamiento de los datos empresariales. Por ello, la migración hacia MongoDB busca resolver estas dificultades y mejorar la toma de decisiones dentro de GlobalMarket mediante un sistema de datos más avanzado y orientado al análisis.

## Requerimientos del sistema

### Software

- MongoDB Atlas
- MongoDB Compass
- Navegador web
- Editor de texto (opcional)

### Hardware

- Computador con Windows
- Conexión a Internet

# Modelado y Validación

Diagrama relacional:

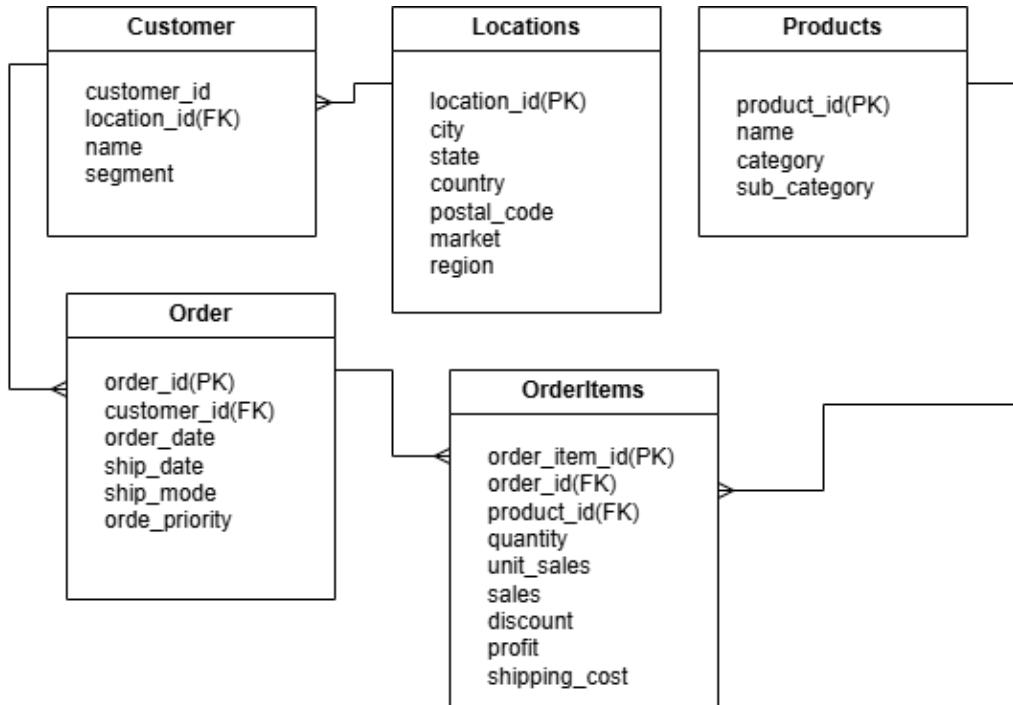
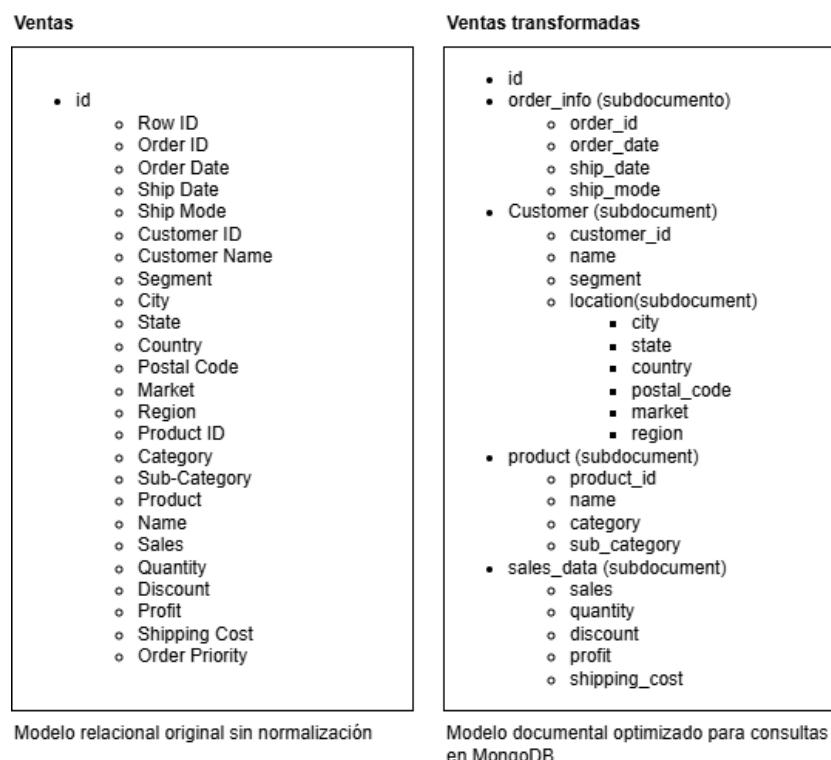


Diagrama documental:



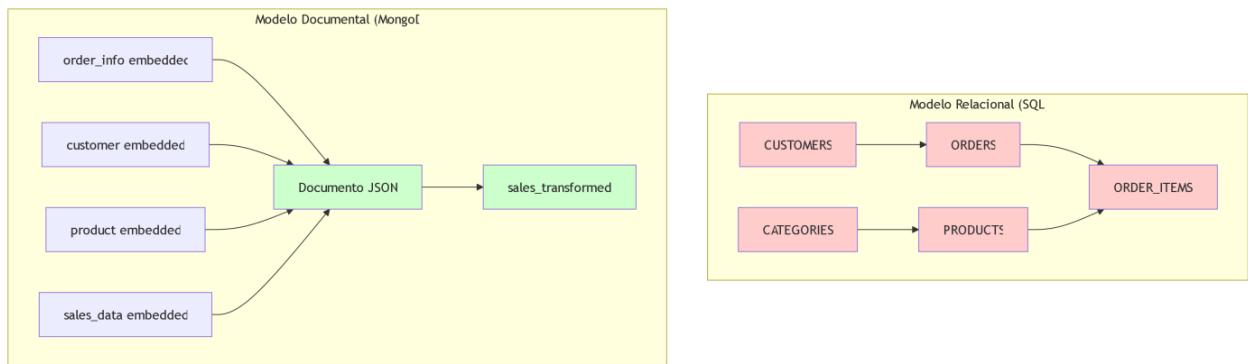
Modelo relacional original sin normalización

Modelo documental optimizado para consultas en MongoDB

La base de datos relacional garantiza integridad y normalización, pero implica un mayor costo en consultas analíticas debido a la fragmentación de la información en múltiples tablas.

La base de datos documental desarrollada en MongoDB permite un acceso más ágil a información consolidada por pedido, cliente y producto, lo cual beneficia directamente los análisis operativos y estratégicos del negocio GlobalMarket.

Por esa razón, la migración a un modelo NoSQL documental representa una mejora clara en eficiencia, escalabilidad y velocidad para el manejo de grandes volúmenes de datos comerciales.



# Desarrollo

## 1. Configuración del entorno en MongoDB Atlas

### a. Crear el GlobalMarket-Cluster en Mongo DB atlas

**Connect to GlobalMarket-Cluster**

1 Set up connection security      2 Choose a connection method      3 Connect

You need to secure your MongoDB Atlas cluster before you can use it. Set which users and IP addresses can access your cluster now. [Read more](#)

I. Add a connection IP address

Your current IP address [45.186.208.72] has been added to enable local connectivity. Only an IP address you add to your Access List will be able to connect to your project's clusters. Add more later in [Network Access](#).

II. Create a database user

A database user has been added to this project. Create another user later in [Database Access](#). You'll need your database user's credentials in the next step.

**Choose a connection method**

1 Set up connection security      2 Choose a connection method      3 Connect

**Connect to your application**

 Drivers  
Access your Atlas data using MongoDB's native drivers (e.g. Node.js, Go, etc.)

**Access your data through tools**

 Compass  
Explore, modify, and visualize your data with MongoDB's GUI

 Shell  
Quickly add & update data using MongoDB's Javascript command-line interface

 MongoDB for VS Code  
Work with your data in MongoDB directly from your VS Code environment

 Atlas SQL  
Easily connect SQL tools to Atlas for data analysis and visualization

 Model Context Protocol (MCP) Server  
Access your data in agentic developer tools (Claude, Cursor, VS Code, Windsurf)

**Go Back** **Close**

## b. Copiar el String

Connect to GlobalMarket-Cluster

1. Set up connection security    2. Choose a connection method    3. Connect

Connecting with MongoDB Compass

I don't have MongoDB Compass installed    I have MongoDB Compass installed

1. Select your operating system and download MongoDB Compass

macOS x64 (Intel) (11+)

Download Compass (1.48.2) or Copy download URL

Compass is an interactive tool for querying, optimizing, and analyzing your MongoDB data.

2. Copy the connection string, then open MongoDB Compass

Show Password ⓘ

Use this connection string in your application

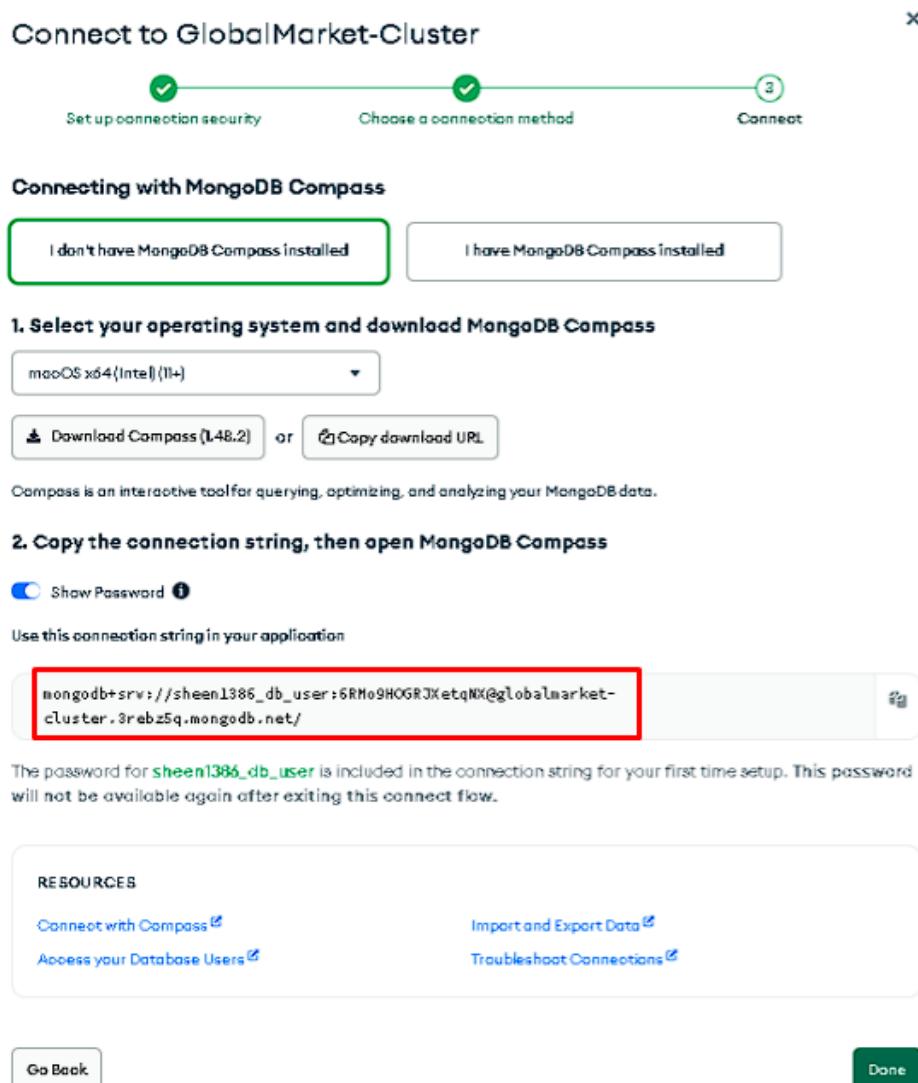
`mongodb+srv://sheen1386_db_user:5RMo9HOGRJXetqN@globalmarket-cluster.3rebz5q.mongodb.net/`

The password for `sheen1386_db_user` is included in the connection string for your first time setup. This password will not be available again after exiting this connect flow.

RESOURCES

Connect with Compass ⓘ    Import and Export Data ⓘ  
Access your Database Users ⓘ    Troubleshoot Connections ⓘ

Go Back    Done



## c. Pegar la URL en el MongoDB Compass

New Connection

Manage your connection settings

URI ⓘ    Edit Connection String ⓘ

`mongodb+srv://sheen1386_db_user:*****@globalmarket-cluster.3rebz5q.mongodb.net/`

Name: globalmarket-cluster.3rebz5q.mongodb.net    Color: Purple

Favorite this connection

How do I find my connection string in Atlas?

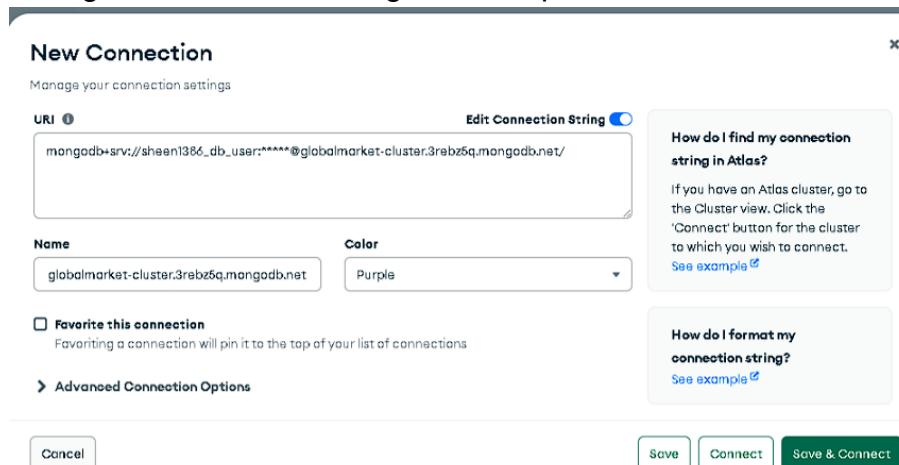
If you have an Atlas cluster, go to the Cluster view. Click the 'Connect' button for the cluster to which you wish to connect.  
See example ⓘ

How do I format my connection string?

See example ⓘ

Advanced Connection Options

Cancel    Save    Connect    Save & Connect



## 2. Conexión del cliente MongoDB Compass

### a. Descarga del archivo CSV

Global Super Store Dataset

Data Card Code (19) Discussion (4) Suggestions (0)

Global\_Superstore2.csv (12.09 MB)

Detail Compact Column

Row ID Order ID Order Date Ship Date Ship Mode Custom

Row ID	Order ID	Order Date	Ship Date	Ship Mode	Custom
1	25035 unique values	1430 unique values	1464 unique values	Standard Class Second Class Other (10206)	60% 20% 20%
48155	CA-2814-135989	14-10-2014	21-10-2014	Standard Class	JW-15228
48936	CA-2812-116638	28-01-2012	31-01-2012	Second Class	JH-15985
34577	CA-2811-182988	05-04-2011	09-04-2011	Second Class	GM-14695
28879	ID-2012-28482	19-04-2012	22-04-2012	First Class	AJ-10788
45794	SA-2011-1838	27-12-2011	29-12-2011	Second Class	MM-7268
4132	MX-2012-130015	13-11-2012	13-11-2012	Same Day	VF-21715
27784	IN-2013-73951	06-06-2013	08-06-2013	Second Class	PF-19120

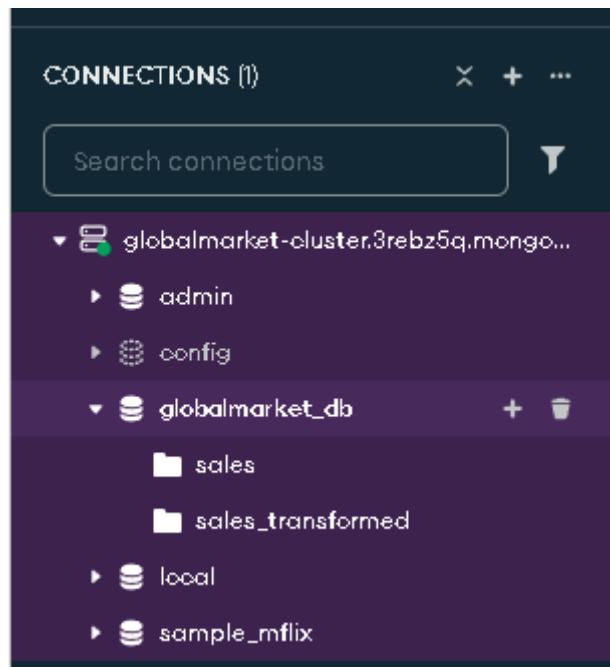
Version 2 (20.13 MB)

Global\_Superstore2.csv  
Global\_Superstore2.xlsx

Summary

2 files  
48 columns

### b. Creación de la base de datos **globalmarket\_dv** en el Compass y el nombre de la colección **Sales**



c. Importamos el archivo (csv)

The screenshot shows the MongoDB Compass 'Import' dialog. At the top, it says 'To collection globalmarket\_db.sales'. Below that, 'Import file:' is set to 'Global\_Superstore2.csv'. Under 'Options', 'Select delimiter' is set to 'Comma', and 'Ignore empty strings' is checked. There is also an unchecked option 'Stop on errors'. The 'Specify Fields and Types' section shows the schema for the 'sales' collection:

Row ID	Order ID	Order Date	Ship Date	Ship Mode	Customer
32298	CA-2012-124891	31-07-2012	31-07-2012	Same Day	RH-19495
26341	IN-2013-77878	06-02-2013	07-02-2013	Second Class	JR-16210
25330	IN-2013-71249	17-10-2013	18-10-2013	First Class	CR-12730
13624	ES-2013-1679342	28-01-2013	30-01-2013	First Class	KM-16375

At the bottom right are 'Cancel' and 'Import' buttons.

d. Data cargada

The screenshot shows the MongoDB Compass interface with the 'sales' collection selected. The left sidebar shows connections and the current connection 'globalmarket-cluster.3rebz5q.mongodb.net'. The main area displays the 'Documents' tab for the 'sales' collection, which contains 51K documents. A search bar at the top says 'Type a query: { field: 'value' } or Generate query'. Below the search bar are buttons for 'ADD DATA', 'EXPORT DATA', 'UPDATE', and 'DELETE'. The document details pane shows a single document's fields:

```

_id: ObjectId('692d135d209add783aae2dca')
Row ID : 32298
Order ID : "CA-2012-124891"
Order Date : "31-07-2012"
Ship Date : "31-07-2012"
Ship Mode : "Same Day"
Customer ID : "RH-19495"
Customer Name : "Rick Hansen"
Segment : "Consumer"
City : "New York City"
State : "New York"
Country : "United States"
Postal Code : 10024
Market : "US"
Region : "East"
Product ID : "TEC-AC-10003033"
Category : "Technology"
Sub-Category : "Accessories"
Product Name : "Plantronics CS510 - Over-the-Head monaural Wireless Headset System"
Sales : 2309.65
Quantity : 7
Discount : 0
Profit : 762.1945
Shipping Cost : 933.57
Order Priority : "Critical"
  
```

Below this, another document's fields are partially visible:

```

_id: ObjectId('692d135d209add783aae2dcb')
Row ID : 26341
  
```

### **3.Implementación del Schema Validation**

#### a. Validación

```
{  
  "$jsonSchema": {  
    "bsonType": "object",  
    "required": ["order_info", "customer", "product", "sales_data"],  
    "properties": {  
      "sales_data.sales": {  
        "bsonType": "double",  
        "minimum": 0,  
        "description": "Sales must be a positive number"  
      },  
      "sales_data.profit": {  
        "bsonType": "double",  
        "description": "Profit must be a number (can be negative)"  
      },  
      "sales_data.quantity": {  
        "bsonType": "int",  
        "minimum": 1,  
        "description": "Quantity must be positive integer"  
      },  
      "sales_data.discount": {  
        "bsonType": "double",  
        "minimum": 0,  
        "maximum": 1,  
        "description": "Discount must be between 0 and 1"  
      },  
    }  
  }  
}
```

```

"customer.customer_id": {
    "bsonType": "string",
    "description": "Customer ID is required"
},
"product.category": {
    "bsonType": "string",
    "enum": ["Technology", "Furniture", "Office Supplies"],
    "description": "Category must be one of the allowed values"
}
}
}
}
}
}

```

The screenshot shows the Compass MongoDB interface. On the left, there's a sidebar with 'Connections' (test Conexion), 'My Queries', and 'Data Modeling'. Below that is a 'CONNECTIONS' section with a dropdown menu. The main area shows a database structure with 'globalmarket-cluster.3rebz5q.mongodb.net > globalmarket\_db > sales'. The 'Validation' tab is selected. In the center, there's a code editor with the following JSON schema:

```

1  /*
2   * This is a starter template for a schema validation rule for a collection.
3   * More information on schema validation rules can be found at:
4   * https://www.mongodb.com/docs/manual/core/schema-validation/
5   */
6  [
7      "$jsonSchema": {
8          "title": "Library.books",
9          "bsonType": "object",
10         "required": ["fieldname1", "fieldname2"],
11         "properties": {
12             "fieldname1": {
13                 "bsonType": "string",
14                 "description": "Fieldname1 must be a string",
15             },
16             "fieldname2": {
17                 "bsonType": "int",
18                 "description": "Fieldname2 must be an integer",
19             },
20         },
21         "arrayFieldName": {
22             "bsonType": "array",
23             "items": {
24                 "bsonType": "string"
25             },
26             "description": "arrayFieldName must be an array of strings"
27         }
28     }
29 }

```

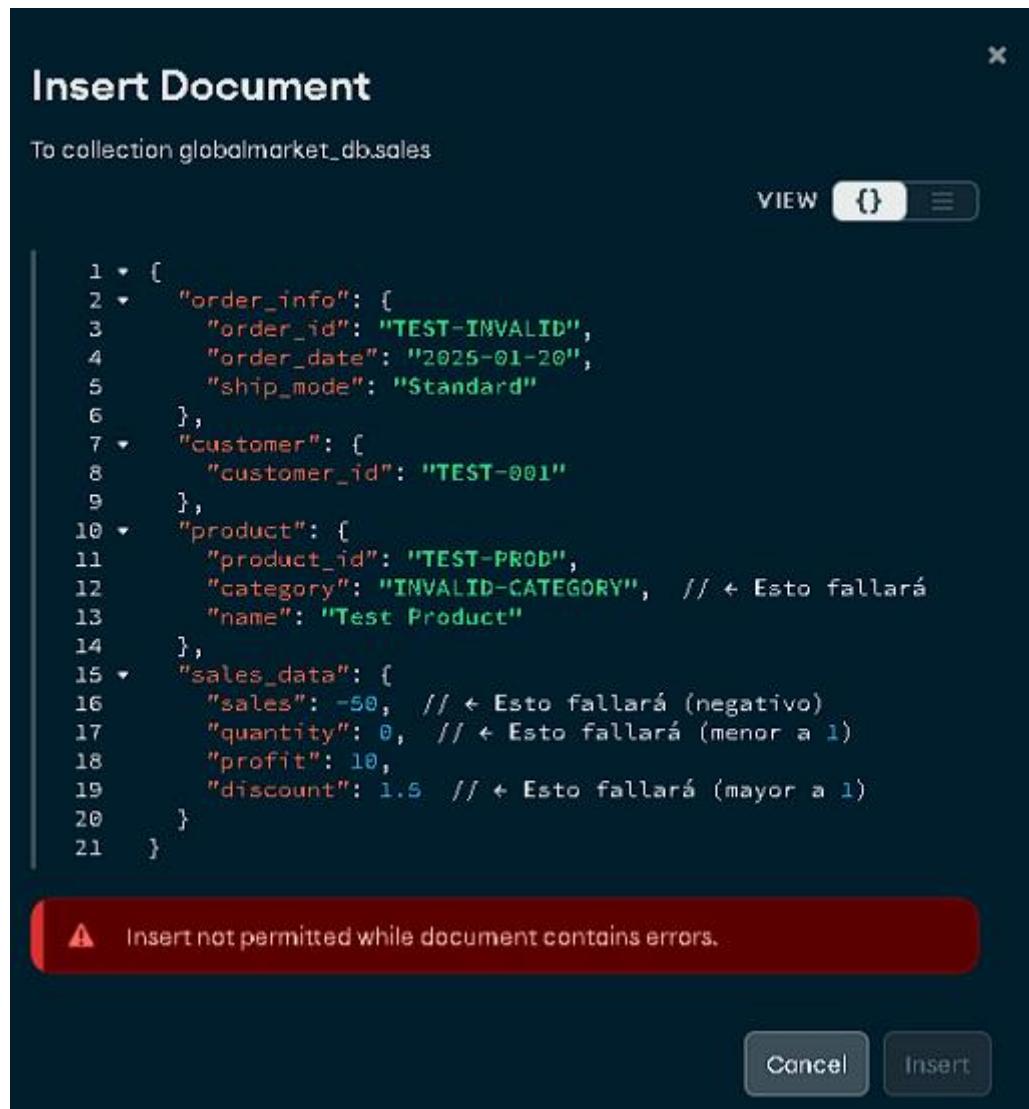
At the bottom of the code editor, a note says: 'Rules are modified & not applied. Please review before applying.' There are 'Cancel' and 'Apply' buttons at the bottom right.

## b. Prueba de validación

```
{
    "order_info": {

```

```
"order_id": "TEST-INVALID",
"order_date": "2025-01-20",
"ship_mode": "Standard"
},
"customer": {
"customer_id": "TEST-001"
},
"product": {
"product_id": "TEST-PROD",
"category": "INVALID-CATEGORY", // ← Esto fallará
"name": "Test Product"
},
"sales_data": {
"sales": -50, // ← Esto fallará (negativo)
"quantity": 0, // ← Esto fallará (menor a 1)
"profit": 10,
"discount": 1.5 // ← Esto fallará (mayor a 1)
}
}
```



## 4. Aplicar validación en sales\_transformed

### a. Validation en sale\_trans

```
{
  "$jsonSchema": {
    "bsonType": "object",
    "required": ["order_info", "customer", "product", "sales_data"],
    "properties": {
      "sales_data.sales": {
```

```
"bsonType": "double",
"minimum": 0,
"description": "Sales must be a positive number"
},
"sales_data.profit": {
  "bsonType": "double",
  "description": "Profit must be a number (can be negative)"
},
"sales_data.quantity": {
  "bsonType": "int",
  "minimum": 1,
  "description": "Quantity must be positive integer"
},
"sales_data.discount": {
  "bsonType": "double",
  "minimum": 0,
  "maximum": 1,
  "description": "Discount must be between 0 and 1"
},
"customer.customer_id": {
  "bsonType": "string",
  "description": "Customer ID is required"
},
"product.category": {
  "bsonType": "string",
  "enum": ["Technology", "Furniture", "Office Supplies"],
  "description": "Category must be one of the allowed values"
}
```

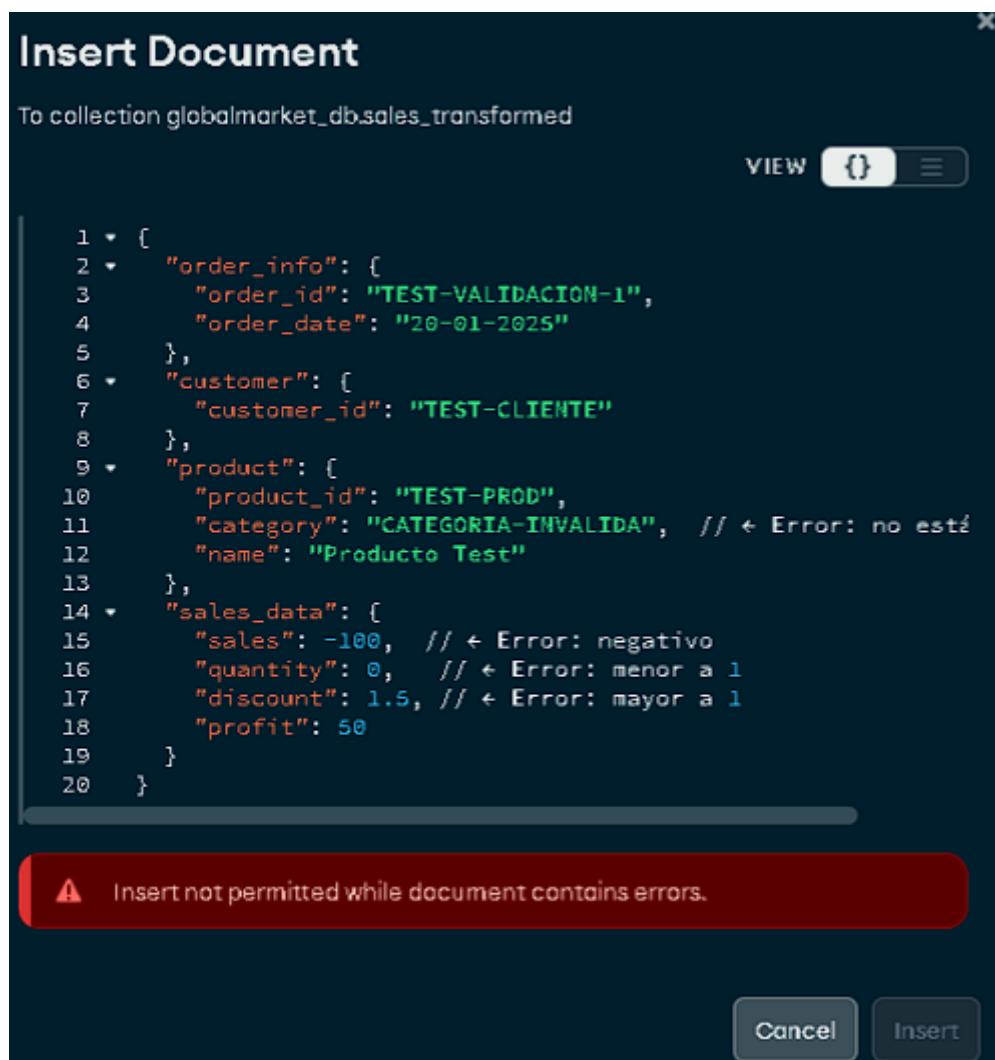
```
    }
}
}
}
```

## b. DATOS DE PRUEBA PARA VALIDACIÓN

### Documento INVÁLIDO (debe fallar):

```
{
  "order_info": {
    "order_id": "TEST-VALIDACION-1",
    "order_date": "20-01-2025"
  },
  "customer": {
    "customer_id": "TEST-CLIENTE"
  },
  "product": {
    "product_id": "TEST-PROD",
    "category": "CATEGORIA-INVALIDA", // ← Error: no está en enum
    "name": "Producto Test"
  },
  "sales_data": {
    "sales": -100, // ← Error: negativo
    "quantity": 0, // ← Error: menor a 1
    "discount": 1.5, // ← Error: mayor a 1
    "profit": 50
  }
}
```

}



c. Documento VÁLIDO (debe pasar):

{

```
"order_info": {  
  "order_id": "TEST-VALIDO-001",  
  "order_date": "20-01-2025",  
  "ship_mode": "Standard",  
  "order_priority": "High"
```

```
        },  
  
        "customer": {  
  
            "customer_id": "CLIENTE-OK",  
  
            "name": "Cliente Prueba",  
  
            "segment": "Consumer",  
  
            "location": {  
  
                "city": "Test City",  
  
                "country": "Test Country"  
  
            }  
  
        },  
  
        "product": {  
  
            "product_id": "PROD-VALIDO",  
  
            "category": "Technology",  
  
            "sub_category": "Accessories",  
  
            "name": "Producto Válido"  
  
        },  
  
        "sales_data": {  
  
            "sales": 250.75,  
  
            "quantity": 3,  
  
            "discount": 0.1,  
  
            "profit": 50.25,  
  
            "shipping_cost": 15.50  
  
        }  
  
    }  

```

Insert Document

To collection globalmarket\_db.sales\_transformed

```
3     "order_id": "TEST-VALIDO-001",
4     "order_date": "20-01-2025",
5     "ship_mode": "Standard",
6     "order_priority": "High"
7   },
8   "customer": {
9     "customer_id": "CLIENTE-OK",
10    "name": "Cliente Prueba",
11    "segment": "Consumer",
12   "location": [
13     "city": "Test City",
14     "country": "Test Country"
15   ],
16 },
17 "product": [
18   "product_id": "PROD-VALIDO",
19   "category": "Technology",
20   "sub_category": "Accessories",
21   "name": "Producto Válido"
22 ],
23 "sales_data": [
24   "sales": 250.75,
25   "quantity": 3,
26   "discount": 0.1,
27   "profit": 50.25,
28   "shipping_cost": 15.50
29 ],
30 }
```

Cancel Insert

## 5. Aplicación del Pipeline de Transformación en Sales

### a. Pipeline en text

```
{
  "$project": {
    "order_info": {
      "order_id": "$Order ID",
      "order_date": "$Order Date",
      "ship_date": "$Ship Date",
      "ship_mode": "$Ship Mode",
```

```
    "order_priority": "$Order Priority"  
},  
  
    "customer": {  
        "customer_id": "$Customer ID",  
        "name": "$Customer Name",  
        "segment": "$Segment",  
        "location": {  
            "city": "$City",  
            "state": "$State",  
            "country": "$Country",  
            "postal_code": "$Postal Code",  
            "market": "$Market",  
            "region": "$Region"  
        }  
},  
  
    "product": {  
        "product_id": "$Product ID",  
        "name": "$Product Name",  
        "category": "$Category",  
        "sub_category": "$Sub-Category"  
},  
  
    "sales_data": {  
        "sales": "$Sales",  
        "quantity": "$Quantity",  
        "discount": "$Discount",  
        "profit": "$Profit",  
        "shipping_cost": "$Shipping Cost"
```

```
}
```

```
}
```

```
}
```

The screenshot shows the MongoDB Compass interface with the following details:

- Left Sidebar (Compass):** Shows connections, queries, and data modeling.
- Top Bar:** MongoDB Compass - globalmarket-cluster.3rebz5q.mongodb.net/globalmarket\_db.sales. Includes tabs for Connections, Edit, View, Collection, Help, and a search bar.
- Central Area:**
  - Collection:** sales
  - Database:** globalmarket\_db
  - Collection:** sales
  - Aggregation Tab:** Active tab.
  - Preview Area:** Shows a sample of 10 documents from the pipeline output. Each document includes fields: \_id, order\_info, customer, product, and sales\_data.
  - Output Options:** Buttons for PREVIEW, STAGES, TEXT, WIZARD, and other settings.
- Code Editor (Bottom Left):**

```

10      },
11      "customer": {
12        "_id": "$Customer_ID",
13        "name": "$Customer_Name",
14        "segment": "$Segment",
15        "location": [
16          {"city": "$City",
17           "state": "$State",
18           "country": "$Country",
19           "postal_code": "$Postal_Code",
20           "market": "$Market",
21           "region": "$Region"
22         }
23       },
24       "product": [
25         {"product_id": "$Product_ID",
26          "name": "$Product_Name",
27          "category": "$Category",
28          "sub_category": "$Sub-Category"
29        },
30        "sales_data": [
31          {"sales": "$Sales",
32           "quantity": "$Quantity",
33           "discount": "$Discount",
34           "profit": "$Profit",
35           "shipping_cost": "$Shipping_Cost"
36         }
37       ]
38     }
39   ]

```

## 6. PIPELINE 1: REPORTE DE VENTAS POR CATEGORÍA Y MES

```
[
{
  $match: {
    "sales_data.sales": { $gt: 0 },      // Solo ventas positivas
    "product.category": { $exists: true } // Categoría definida
  }
},
{
  $addFields: {
    order_month: {

```

```
$dateToString: {  
    format: "%Y-%m",  
    date: { $dateFromString: { dateString: "$order_info.order_date", format: "%d-%m-%Y" } }  
}  
}  
}  
}  
}  
{$group: {  
    _id: {  
        category: "$product.category",  
        month: "$order_month"  
    },  
    total_sales: { $sum: "$sales_data.sales" },  
    total_orders: { $sum: 1 },  
    avg_profit: { $avg: "$sales_data.profit" }  
}  
},  
{$project: {  
    _id: 0,  
    category: "$_id.category",  
    month: "$_id.month",  
    total_sales: 1,  
    total_orders: 1,  
    avg_profit: 1,  
}}
```

```

sales_per_order: { $divide: ["$total_sales", "$total_orders"] }

}

}

{

$sort: { month: 1, total_sales: -1 }

}

]

```

The screenshot shows the MongoDB Compass interface with the following details:

- Collection:** sales\_transformed
- Database:** globalmarket\_db
- Aggregation Pipeline:**
  - \$match
  - \$addFields
  - \$group
  - \$project
  - \$sort
- Results:** Three documents are displayed, each representing a category and its sales statistics for January 2011.

Category	total_sales	total_orders	avg_profit	month	sales_per_order
Furniture	34463.752	80	8.640611249999997	"2011-01"	430.7969124999996
Office Supplies	38526.7275	286	15.992091258741258	"2011-01"	117.22631993006993
Technology	30908.00936	67	45.62498537313433	"2011-01"	461.31355761194027



## 7. Pipeline completo y auto-contenido (genera rating aleatorio y luego filtra):

Como el dataset de ventas no incluye ratings, simulamos un campo rating aleatorio entre 1 y 5 para demostrar el uso del aggregation pipeline con \$match, \$group y \$project, cumpliendo la estructura solicitada.

```
top_productos_mejor_calificacion
```

```
[  
 {  
   $addFields: {  
     rating: {  
       $add: [  
         { $floor: { $multiply: [ { $rand: {} }, 5 ] } },  
         1  
       ]  
     },  
     factor_resenas: 10  
   }  
 },  
 {  
   $group: {  
     _id: "$product.product_id",  
     nombre: { $first: "$product.name" },  
     categoria: { $first: "$product.category" },  
     total_resenas: { $sum: "$factor_resenas" },  
     calificacion_promedio: { $avg: "$rating" }  
   }  
 },  
 {
```

```
$match: {
    total_resenas: { $gt: 50 }
}
},
{
$sort: {
    calificacion_promedio: -1
}
},
{
$project: {
    _id: 0,
    producto_id: "$_id",
    nombre_producto: "$nombre",
    categoria: "$categoria",
    total_resenas: 1,
    calificacion_promedio: { $round: ["$calificacion_promedio", 2] }
}
},
{
$limit: 10
}
]
```

Untitled - modified SAVE + CREATE NEW EXPORT TO LANGUAGE

PREVIEW STAGES TEXT WIZARD

OUTPUT OPTIONS

```

4   rating: {
5     $add: [
6       { $Floor: { $Multiply: [ { $rand: {} }, 5 ] } },
7       1
8     ]
9   },
10  Factor_resenas: 10
11 },
12 },
13 {
14   $group: {
15     _id: "$product.product_id",
16     nombre_producto: "$product.name",
17     categoria: "$product.category",
18     total_resenas: [ $sum: "$factor_resenas" ],
19     calificacion_promedio: [ $avg: "$rating" ]
20   },
21 },
22 {
23   $match: {
24     total_resenas: { $gt: 50 }
25   },
26 },
27 {
28   $sort: { calificacion_promedio: -1 }
29 },
30 {
31   $project: {
32     _id: 0,
33     producto_id: "$_id",
34     nombre_producto: "$nombre",
35     categoria: "$categoria",
36     total_resenas: 1,
37     calificacion_promedio: { $Round: [ "$calificacion_promedio", 2 ] }
38 }
39 }
```

**Pipeline Output Preview**  
Sample of 10 documents

- total\_resenas : 60  
producto\_id : "OFF-PA-10000523"  
nombre\_producto : "Southworth Parchment Paper & Envelopes"  
categoria : "Office Supplies"  
calificacion\_promedio : 4.92
- total\_resenas : 88  
producto\_id : "OFF-FA-10000595"  
nombre\_producto : "DTC Bulk Pack Metal Binder Clips"  
categoria : "Office Supplies"  
calificacion\_promedio : 4.75
- total\_resenas : 68  
producto\_id : "TEC-STA-10000558"  
nombre\_producto : "Startech Card Printer, Red"  
categoria : "Technology"  
calificacion\_promedio : 4.67
- total\_resenas : 68  
producto\_id : "TEC-PH-10000377"  
nombre\_producto : "Samsung Speaker Phone, with Caller ID"  
categoria : "Technology"  
calificacion\_promedio : 4.67

**My Queries** +

Search All databases

**AGGREGATE**  
**reporte\_ventas\_categ...**  
globalmarket\_db  
sales\_transformed  
Last modified: 36 minutes ago

**AGGREGATE**  
**top\_productos\_mejor\_...**  
globalmarket\_db  
sales\_transformed  
Last modified: 2 minutes ago

## 8. Pipeline 3 – Bucket por rangos de precios unitarios

**Objetivo:**

Agrupar productos en 3 rangos de precio unitario: **Bajo, Medio, Alto.**

[

// Paso 1: Calcular precio unitario (sales / quantity)

{

\$addFields: {

precio\_unitario: {

\$cond: {

```

        if: { $gt: ["$sales_data.quantity", 0] },
        then: { $divide: ["$sales_data.sales", "$sales_data.quantity"] },
        else: 0
    }
}
},
},
},
// Paso 2: Agrupar en rangos usando $bucket
{
$bucket: {
    groupBy: "$precio_unitario",
    boundaries: [0, 50, 200, 1000], // Rangos: Bajo (0-50), Medio (50-200), Alto (200-1000)
    default: "Fuera de rango",
    output: {
        count: { $sum: 1 },
        productos: { $push: "$product.name" },
        precio_promedio: { $avg: "$precio_unitario" }
    }
},
},
// Paso 3: Renombrar rangos para claridad
{
$project: {
    _id: 0,
    rango: {
        $switch: {

```

```
branches: [
    { case: { $eq: ["$_id", "Fuera de rango"] }, then: "Fuera de rango" },
    { case: { $lt: ["$_id", 50] }, then: "Bajo" },
    { case: { $lt: ["$_id", 200] }, then: "Medio" },
    { case: { $gte: ["$_id", 200] }, then: "Alto" }
],
default: "No clasificado"
}
},
cantidad_productos: "$count",
precio_promedio: { $round: ["$precio_promedio", 2] }
}
},
// Paso 4: Ordenar por rango
{
$sort: {
rango: 1
}
}
]
```

Untitled - modified SAVE + CREATE NEW EXPORT TO LANGUAGE

```

1 * [
2   // Paso 1: Calcular precio unitario (sales / quantity)
3   [
4     $addFields: [
5       Precio_unitario: [
6         $cond: [
7           if: { $gt: ["$sales_data.quantity", 0] },
8           then: { $divide: ["$sales_data.sales", "$sales_data.quantity"] },
9           else: 0
10        }
11      ]
12    ],
13  ],
14  // Paso 2: Agrupar en rangos usando $bucket
15  [
16    $bucket: [
17      groupBy: "$precio_unitario",
18      boundaries: [0, 50, 200, 1000], // Rangos: Bajo (0-50), Medio (50-200), Alto (200+)
19      default: "Fuera de rango",
20      output: [
21        count: { $sum: 1 },
22        productos: { $push: "$product.name" },
23        precio_promedio: { $avg: "$precio_unitario" }
24      ]
25    ],
26  ],
27  // Paso 3: Renombrar rangos para claridad
28  [
29    $project: [
30      _id: 0,
31      rango: [
32        $switch: [
33          branches: [
34            { case: { $eq: ["$_id", "Fuera de rango"] }, then: "Fuera de rango" },
35            { case: { $lt: ["$_id", 50] }, then: "Bajo" },
36            { case: { $lt: ["$_id", 200] }, then: "Medio" },
37            { case: { $gte: ["$_id", 200] }, then: "Alto" }
38          ],
39          default: "No clasificado"
40        }
41      ],
42      cantidad_productos: "$count",
43    ]
44  ]

```

**PIPELINE OUTPUT PREVIEW**  
Sample of 4 documents

rango : "Alto"	cantidad_productos : 4524	precio_promedio : 345.46
rango : "Bajo"	cantidad_productos : 33199	precio_promedio : 19.18
rango : "Fuera de rango"	cantidad_productos : 37	precio_promedio : 1662.59
rango : "Medio"	cantidad_productos : 13530	precio_promedio : 104.52

## ÍNDICE 1: Compuesto (categoría + fecha)

## Create Index

globalmarket\_db.sales\_transformed

### Index fields

product.product\_id

1(asc)



### ▼ Options

#### Create unique index

A unique index ensures that the indexed fields do not store duplicate values; i.e. enforces uniqueness for the indexed fields.

#### Index name

Enter the name of the index to create, or leave blank to have MongoDB create a default name for the index.

idx\_product\_id

*Optional*

#### Create TTL

TTL indexes are special single-field indexes that MongoDB can use to automatically remove documents from a collection after a certain amount of time or at a specific clock time.

Cancel

Create Index

ÍNDICE 2: Simple (product\_id)

## Create Index

globalmarket\_db.sales\_transformed

**Index fields**

sales_data.sales	1(asc)	+	-
sales_data.quantity	1(asc)	+	-

**Options**

**Create unique index**  
A unique index ensures that the indexed fields do not store duplicate values; i.e. enforces uniqueness for the indexed fields.

**Index name**  
Enter the name of the index to create, or leave blank to have MongoDB create a default name for the index.  
 Optional

**Create TTL**  
TTL indexes are special single-field indexes that MongoDB can use to

**Cancel** **Create Index**

### ÍNDICE 3: Compuesto (sales + quantity)

sales_transformed		globalmarket-cluster.3rebz5q.mongodb.net > globalmarket_db > sales_transformed						Open MongoDB shell	
Documents	5K	Aggregations		Schema		Indexes	5	Validation	
Create		Refresh		VIEWING				INDEXES	SEARCH INDEXES
Name & Definition	Type	Size	Usage	Properties	Status				
> _id_	REGULAR	2.0 MB	47 (since Mon Dec 01 2025)	UNIQUE	READY				
> text_product_name	TEXT	2.2 MB	10 (since Thu Dec 04 2025)		READY				
> idx_category_date	REGULAR	643.1 kB	0 (since Thu Dec 04 2025)	COMPOUND	READY				
> idx_product_id	REGULAR	622.6 kB	0 (since Thu Dec 04 2025)		READY				
> idx_sales_quantity	REGULAR	720.9 kB	0 (since Thu Dec 04 2025)	COMPOUND	READY				

## Observación

En la configuración de la búsqueda con Atlas Search, debíamos poder buscar productos por nombre con tolerancia a errores tipográficos, es decir, con índice Fuzzy\_product\_search en product.name. Debido a las limitaciones del tier M0 gratuito, las consultas Search no retornan resultados.

```
{  
  "mappings": {  
    "dynamic": false,  
    "fields": {  
      "product.name": {  
        "analyzer": "lucene.standard",  
        "searchAnalyzer": "lucene.standard",  
        "type": "string"  
      }  
    }  
  }  
}
```

fuzzy\_product\_search READY QUERYABLE

This search index parses the data in **globalmarket\_db.sales\_transformed** and has the following configurations.

[View Atlas Search Docs](#)

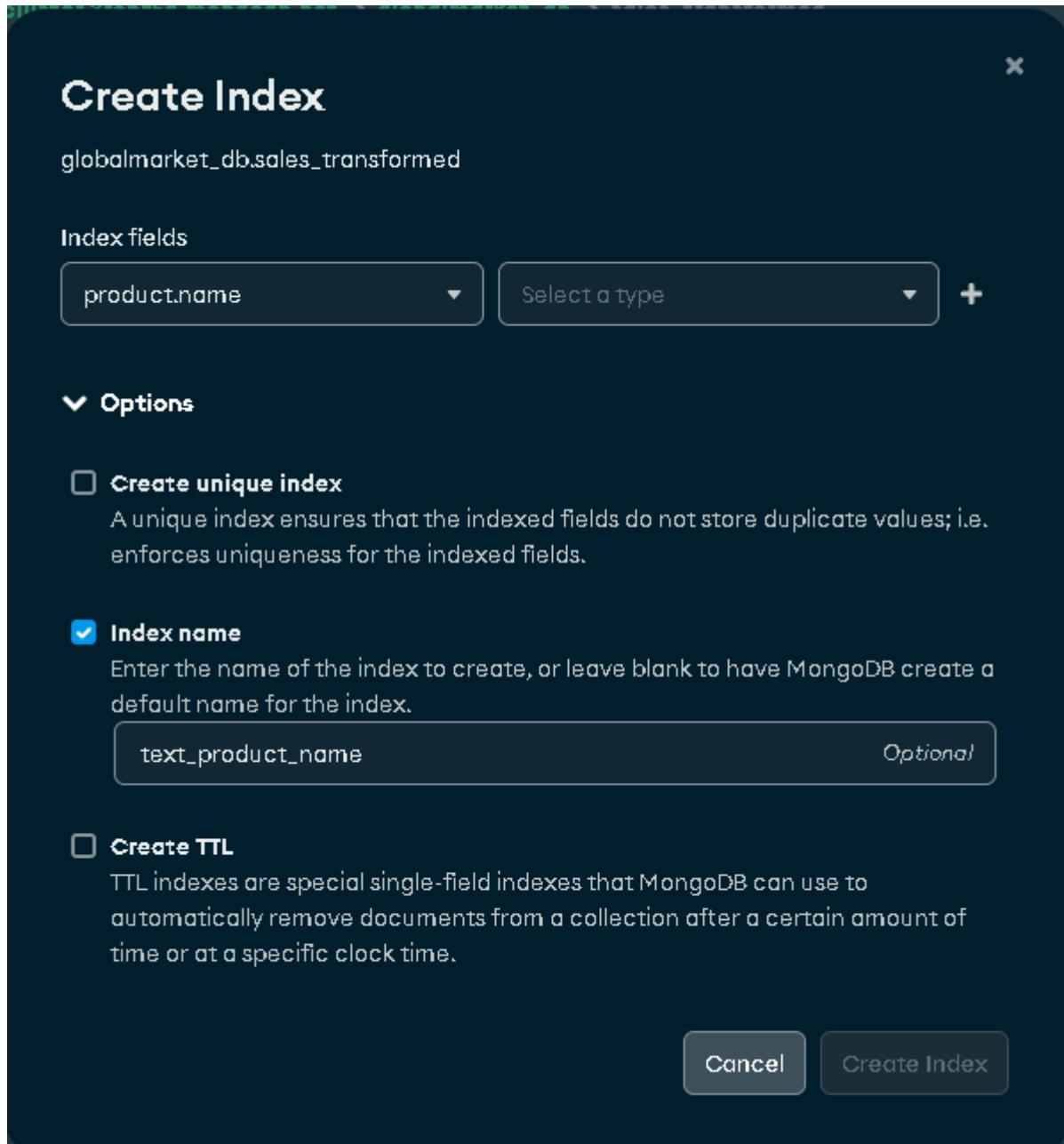
```
1  {  
2    "mappings": {  
3      "dynamic": false,  
4      "fields": {  
5        "product.name": {  
6          "analyzer": "lucene.standard",  
7          "searchAnalyzer": "lucene.standard",  
8          "type": "string"  
9        }  
10       }  
11     }  
12   }
```

Database	Collection	Index Name	Status	Queryable	Type	Index Fields	Documents (Estimated)	Size	Actions
globalmarket_db	sales_transformed	fuzzy_product_search	READY	✓	search	"product.name"	61,290 (100%) indexed of 61,290	783.2KB	<span style="border: 1px solid #4CAF50; border-radius: 15px; padding: 2px 10px; background-color: #c8e6c9;">QUERY</span> ...

[Learn more about Atlas Search](#) [Learn more about Atlas Vector Search](#)

Como alternativa funcional, implementamos un índice de texto tradicional con text y simulamos la búsqueda difusa

Creamos el index



```
[  
{  
  $match: {  
    $text: { $search: "Headset" }  
  }  
},  
{  
  $project: {
```

```

    "product.name": 1,
    "product.category": 1,
    score: { $meta: "textScore" }
}
},
{
$sort: {
    score: { $meta: "textScore" }
}
},
{
$limit: 5
}
]

```

**Pipeline Output Preview**

Sample of 5 documents

**OUTPUT OPTIONS ▾**

```

1  [
2   {
3     $match: {
4       $text: { $search: "Headset" }
5     }
6   },
7   {
8     $project: {
9       "product.name": 1,
10      "product.category": 1,
11      score: { $meta: "textScore" }
12    }
13  },
14  {
15    $sort: {
16      score: { $meta: "textScore" }
17    }
18  },
19  {
20    $limit: 5
21  }
22 ]

```

```

_id: ObjectId('692d1362209add783aae8464')
▶ product : Object
  name : "Samsung Headset, VoIP"
  category : "Technology"
  score : 0.6666666666666666

_id: ObjectId('692d1363209add783aae96fb')
▶ product : Object
  name : "Nokia Headset, Cordless"
  category : "Technology"
  score : 0.6666666666666666

_id: ObjectId('692d1361209add783aae7486')
▶ product : Object
  score : 0.6666666666666666

_id: ObjectId('692d1361209add783aae76df')
▶ product : Object
  score : 0.6666666666666666

```





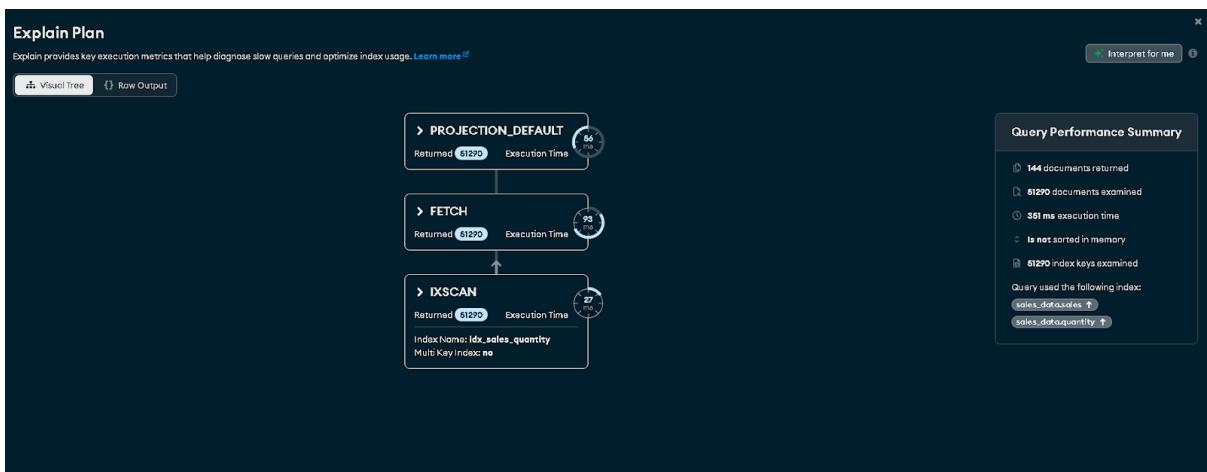
# Explain plan

## Paso 1: Cargar el pipeline

1. Ve a "Aggregations".
2. Haz clic en "Saved Pipelines" (arriba a la izquierda).
3. Selecciona reporte\_ventas\_categoria\_mes (que ya guardaste).

## Paso 2: Ejecutar Explain

4. Arriba a la derecha verás varios botones:  
"Explain" (ícono de gráfica ) | "Export" | "Run" | "Options".
5. Haz clic en "Explain".



Del archivo JSON

Índice usado. **Muestra que se usó un índice** (no COLLSCAN).

```
    "inputStage": {  
        "stage": "IXSCAN",  
        "keyPattern": {  
            "sales_data.sales": 1,  
            "sales_data.quantity": 1  
        },  
        "indexName": "idx_sales_quantity",  
        "isMultiKey": false,  
        "multiKeyPaths": {  
            "sales_data.sales": [],  
            "sales_data.quantity": []  
        }  
    }
```

## Estadísticas de ejecución

- **Tiempo total:** 351 ms.
- **Documentos examinados:** 51,290 (todos los documentos, porque el filtro sales > 0 aplica a casi todos).

```

    "executionSuccess": true,
    "nReturned": 51290,
    "executionTimeMillis": 351,
    "totalKeysExamined": 51290,
    "totalDocsExamined": 51290,
    ...

```

**Plan rechazado. Muestra que MongoDB consideró otro índice (idx\_category\_date) pero eligió idx\_sales\_quantity porque estimó que era más eficiente.**

```

    ...
    "rejectedPlans": [
        {
            "isCached": false,
            "stage": "PROJECTION_DEFAULT",
            "transformBy": [
                "order_info.order_date": 1,
                "product.category": 1,
                "sales_data.profit": 1,
                "sales_data.sales": 1,
                "_id": 0
            ],
            "inputStage": {
                "stage": "FETCH",
                "filter": {
                    "$and": [
                        {
                            "product.category": {
                                "$exists": true
                            }
                        },
                        {
                            "sales_data.sales": {
                                "$gt": 0
                            }
                        }
                    ]
                }
            },
            "inputStage": {
                "stage": "IXSCAN",
                "keyPattern": {
                    "product.category": 1,
                    "order_info.order_date": 1
                },
                "indexName": "idx_category_date",
                "isMultiKey": false,
                "multiKeyPaths": {
                    "product.category": [],
                    "order_info.order_date": []
                }
            }
        }
    ]
}

```

*El Explain Plan muestra que la consulta utilizó el índice idx\_sales\_quantity (IXSCAN) en lugar de un escaneo completo (COLLSCAN), examinando 51,290 claves y tomando 351 ms. MongoDB también consideró el índice idx\_category\_date pero lo rechazó basado en el optimizador de consultas*

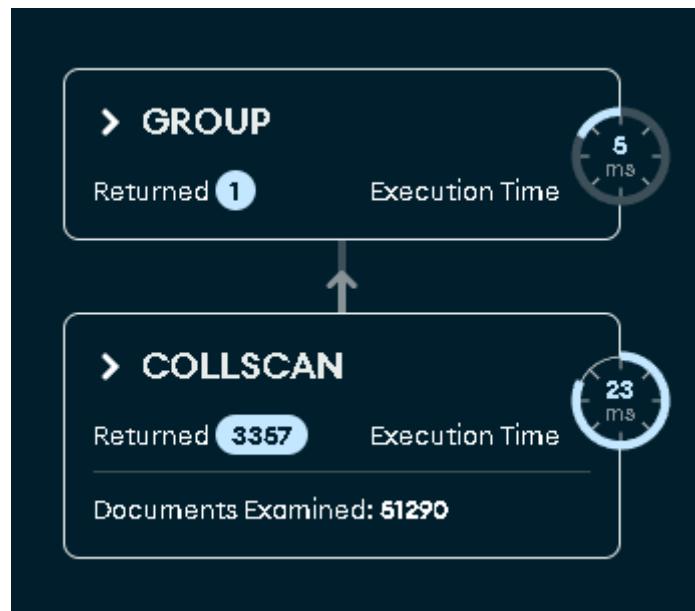
### Una demostración directa

**Sin índice (COLLSCAN):**

[

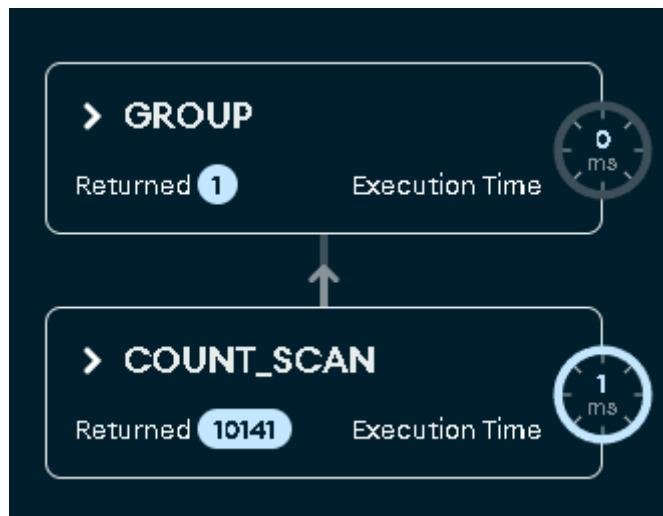
```
{ $match: { "product.sub_category": "Phones" } }, // Campo sin índice
```

```
{ $count: "total" }  
]
```



**Con índice (IXSCAN):**

```
[  
  { $match: { "product.category": "Technology" } }, // Usa idx_category_date  
  { $count: "total" }  
]
```



# Visualización



<https://charts.mongodb.com/charts-project-0-xueaodb/public/dashboards/9a480942-1fd4-454e-9ae7-1e0342ac5e5c>

## Dashboard

### 1. Ventas por país (gráfico de barras – arriba izquierda)

Es un ranking de ventas totales por país.

El eje Y muestra la suma de ventas.

El eje X muestra los países.

Estados Unidos aparece como el país con mayores ventas (más de 2.400.000).

Luego vienen Indonesia, México, Honduras, Bangladesh, etc.

La forma de la curva indica que pocos países concentran la mayoría de las ventas, típico comportamiento long tail.

### 2. Cantidad por ganancias (gráfico de rosquilla – arriba centro)

Muestra la distribución de la ganancia agrupada por rangos.

Cada color representa un rango de ganancia (ej: 500–550, 600–650, etc.).

La gráfica permite ver cuáles rangos de ganancia son más frecuentes.

Parece que los rangos medios (alrededor de 300–450) concentran más cantidad.

### **3. Productos vendidos (gráfico horizontal – arriba derecha)**

Compara dos métricas:

Cantidad de ventas (verde)

Cantidad de productos distintos vendidos (azul)

Las barras están normalizadas (0–100%).

Se observan productos específicos como:

Staples

Eidon Tray

Novimex Stool

Brother Fax

SanDisk Memo Slips

etc.

Permite identificar qué productos tienen más movimiento o más variedad.

### **4. Ventas Totales (parte inferior izquierda)**

Es un KPI destacado que muestra el total de ventas acumuladas.

Valor total de ventas: 12,642,501.9099

Representa todo el ingreso comercial sumado en la base de datos.

### **5. Productos por fecha — Productos más pedidos por año (abajo centro)**

Gráfico lineal de número de pedidos por fecha.

En el eje X aparecen fechas ordenadas.

En el eje Y se muestra el número de pedidos.

La curva es descendente: algunas fechas o años tienen muchos pedidos y luego va cayendo en orden ranking.

Funciona como un top de fechas más demandadas.

## 6. Producto por país – Mapa mundial (abajo derecha)

Mapa de calor que muestra cuántos productos se venden por país.

Tonos más oscuros = mayor cantidad.

Estados Unidos aparece con el mayor conteo (al nivel de 9000 productos).

Otros países tienen valores más bajos.

Permite ver dónde hay mayor actividad comercial.

## Anexo:

Repositorio de Github <https://github.com/haddan17/GlobalMarket---MongoDB>

## Recomendaciones generales

- Fortalecer la validación de datos configurando reglas estrictas para tipos, rangos y valores permitidos, de forma que la base de datos rechace automáticamente registros inconsistentes y mantenga la calidad de la información.
- Incorporar reglas de negocio adicionales en el esquema (por ejemplo, coherencia entre ventas, cantidad, descuentos y precios unitarios) para asegurar que los datos utilizados en los reportes reflejen correctamente la realidad comercial.
- Optimizar los pipelines de agregación colocando los filtros y proyecciones en las primeras etapas y verificando que los campos usados en filtros, agrupaciones y ordenamientos estén debidamente indexados, con el fin de reducir documentos examinados y mejorar los tiempos de respuesta.
- Evaluar el uso de un plan de servicio superior en la nube cuando se requiera búsqueda difusa real (fuzzy search) sobre nombres de productos, aprovechando índices especializados de búsqueda en lugar de depender únicamente de índices de texto tradicionales.
- Enriquecer el dashboard analítico incorporando indicadores accionables (productos de alta y baja rotación, clientes de alto valor, fechas pico de demanda, anomalías en descuentos), de manera que las visualizaciones apoyen decisiones concretas de ventas, marketing e inventario.

## Conclusión

El modelo documental aplicado a través de MongoDB consolida en un solo documento la información de pedido, cliente, producto y datos de venta, lo que reduce la fragmentación típica del modelo relacional y facilita consultas analíticas y operativas sobre ventas, categorías y clientes. Esta estructura, junto con los pipelines de agregación implementados (reportes por categoría y mes, bucket de rangos de precios, ranking de productos, etc.), muestra un uso correcto del framework de agregación para análisis de negocio y soporta un dashboard claro en Charts para monitorear ventas, ganancias y distribución geográfica. Además, el uso de validación con `jsonSchema` y de índices compuestos mejora la calidad de los datos y demuestra preocupación por el rendimiento de las consultas al evitar escaneos completos de colección mediante IXSCAN, lo que es consistente con las buenas prácticas de MongoDB para validación e indexación.