

Proyecto


Diseño e implementación de la automatización de una infraestructura sobre AWS


Lo primero que hacemos es conectarnos a nuestro clúster **iebs** en MongoDB Atlas donde vamos a crear nuestra base de datos **proyecto**.

CLUSTER IEBS > DATA WAREHOUSE

Database Deployments

Find a database deployment...

 Connect View Monitoring Browse Collections ...

 Enhance Your Experience

For production throughput and richer metrics, upgrade to a dedicated cluster now!

Upgrade

R 0

W 0

03/23/23 - 03:26 AM

100.0/s

Connections 0

03/23/23 - 03:26 AM

100.0

VERSION	REGION	CLUSTER TIER	TYPE	BACKUPS	LI
5.0.15	AWS / Paris (eu-west-3)	M0 Sandbox (General)	Replica Set - 3 nodes	Inactive	Ni

Pinchamos en [Create Database](#)

CLUSTER IEBS > DATA WAREHOUSE > DATABASES

iebs

Overview Real Time Metrics Collections Search Profiler Performance Advisor Online Archi

DATABASES: 5 COLLECTIONS: 8

+ Create Database

Search Namespaces

Tienda

Productos

Ventas

console

juegos

prueba_mia

sucursales

Tienda.Productos

STORAGE SIZE: 20KB LOGICAL DATA SIZE: 431B TOTAL DOCUMENTS: 5 INDEXES TOTAL SIZE: 20KB

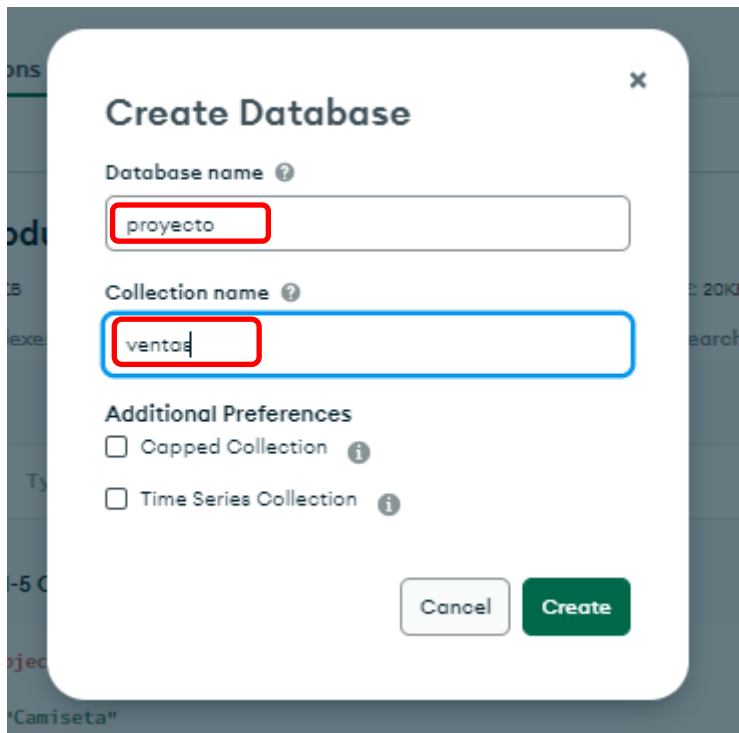
Find Indexes Schema Anti-Patterns Aggregation Search Indexes

Filter Type a query: { field: 'value' }

QUERY RESULTS: 1-5 OF 5

```
{
  "_id": ObjectId("6409d0ef87d1244c979f142e"),
  "id": 1,
  "name": "Camiseta",
  "category": "Camisetas",
  "price": 18.99
}
```

Y creamos la base de datos **proyecto** y la colección **ventas**.



Create Database

Database name [?]

Collection name [?]

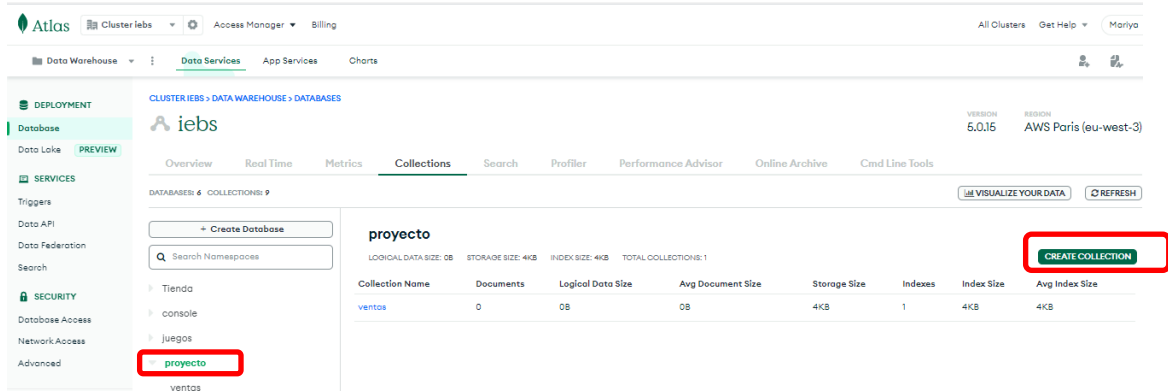
Additional Preferences

☐ Capped Collection ⁱ

☐ Time Series Collection ⁱ

Cancel Create

Una vez creados pinchamos en la base de datos **proyecto** y a la derecha pinchamos en **CREATE COLLECTION** para crear la colección **videojuegos**



Cluster iebes | Data Warehouse | Data Services | App Services | Charts

Database: proyecto

VERSION: 5.0.15 | REGION: AWS Paris (eu-west-3)

Overview | Real Time | Metrics | **Collections** | Search | Profiler | Performance Advisor | Online Archive | Cmd Line Tools

DATABASES: 4 | COLLECTIONS: 1

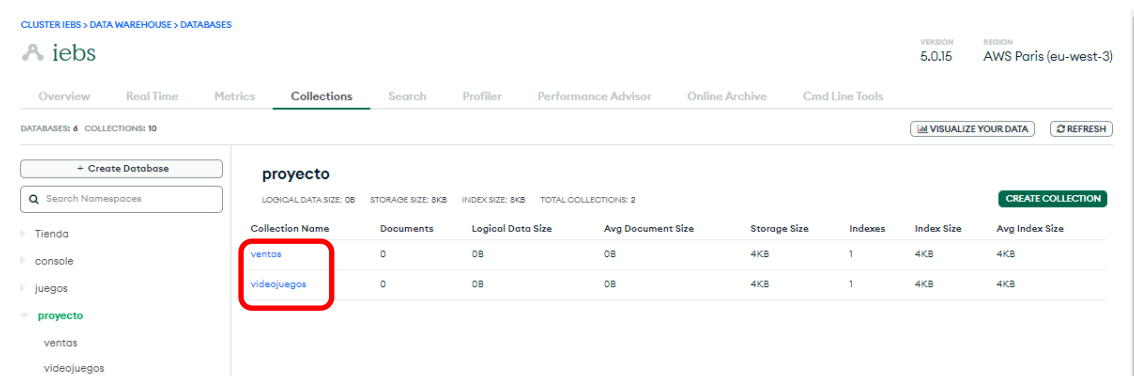
+ Create Database

Search Namespaces

Collection Name	Documents	Logical Data Size	Avg Document Size	Storage Size	Indexes	Index Size	Avg Index Size
ventas	0	0B	0B	4KB	1	4KB	4KB

CREATE COLLECTION

Ya las tenemos creadas.



Cluster iebes | Data Warehouse | Data Services | App Services | Charts

Database: proyecto

VERSION: 5.0.15 | REGION: AWS Paris (eu-west-3)

Overview | Real Time | Metrics | **Collections** | Search | Profiler | Performance Advisor | Online Archive | Cmd Line Tools

DATABASES: 4 | COLLECTIONS: 2

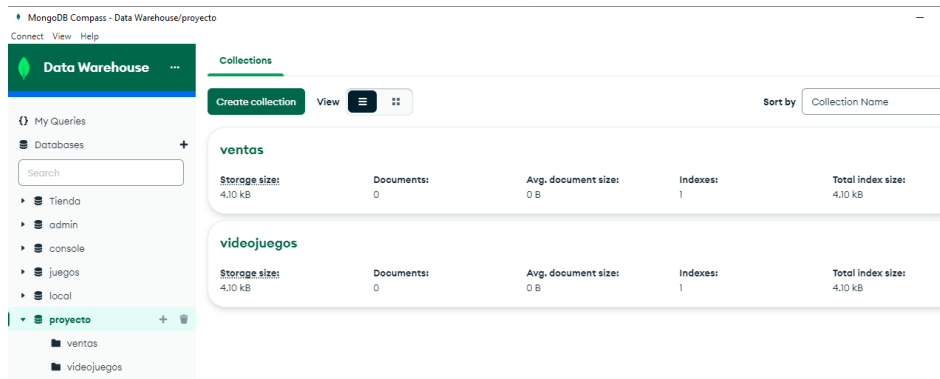
+ Create Database

Search Namespaces

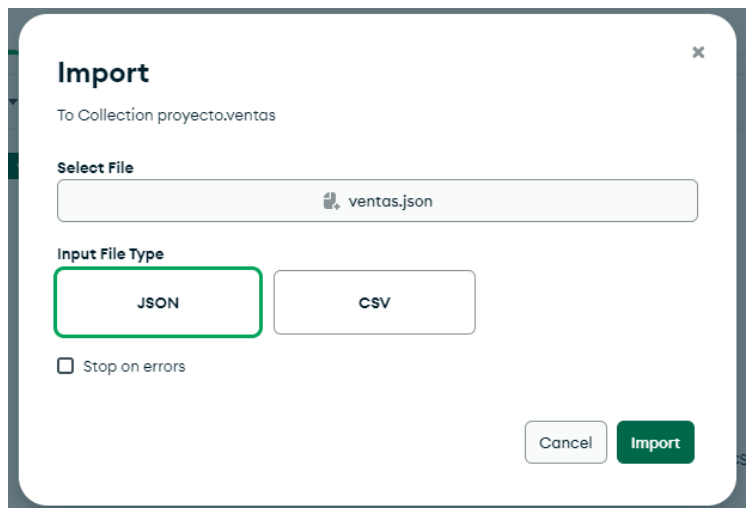
Collection Name	Documents	Logical Data Size	Avg Document Size	Storage Size	Indexes	Index Size	Avg Index Size
ventas	0	0B	0B	4KB	1	4KB	4KB
videojuegos	0	0B	0B	4KB	1	4KB	4KB

CREATE COLLECTION

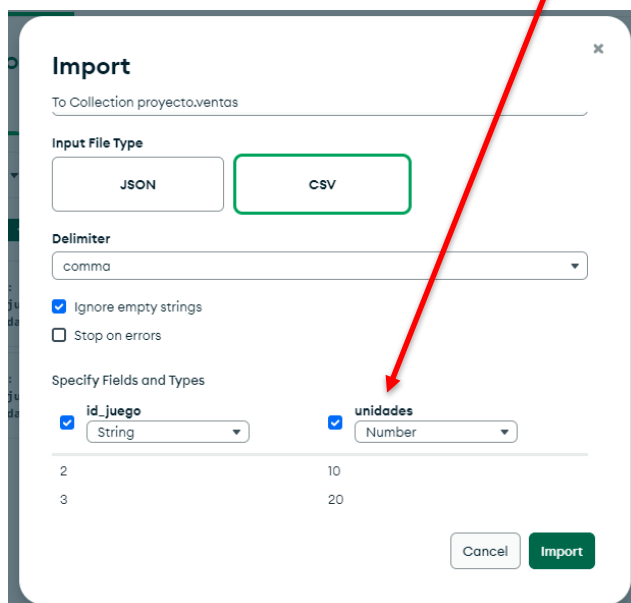
Nos conectamos con **MongoDB Compass** y vemos la base de datos **proyecto** creada en **MongoDB Atlas** reflejada en **MongoDB Compass**.



Importamos los datos. 2 ficheros **json** y 2 ficheros **csv**



Al importar el **csv**, vemos que los datos son de tipo **string**. Aprovechamos la oportunidad de convertir la columna **unidades** en **Integer** ya que tenemos que hacer operaciones.



Vemos que el **csv** que hemos importado los datos de unidades ya son **integer** pero el **json** que importamos son de tipo **string**. Los modificamos ya que son solo dos documentos. En caso de que es un fichero muy grande se puede hacer por código (los pipelines) con **\$toInt**

proyecto.ventas

7 DOCUMENTS 1 INDEXES

Documents Aggregations Schema Explain Plan Indexes Validation

Filter Type a query: { field: 'value' } Reset Find More Options

ADD DATA EXPORT COLLECTION 1 - 7 of 7

```
{
  "_id": ObjectId('641c1afea9395bfa879a1113'),
  "id_juego": "1",
  "unidades": 2
}
```

1 _id: ObjectId('641c1afea9395bfa879a1114') ObjectId
2 id_juego: "5" String
3 unidades: 6 Int64

Document modified. CANCEL UPDATE

```
{
  "_id": ObjectId('641c1bd8a9395bfa879a1115')
}
```

Al poner el tipo de dato **Int64** vemos que no se nos pintan en azul como los demás y nos damos cuenta que el tipo de dato es **Int32**, lo volvemos a modificar.

proyecto.ventas

7 DOCUMENTS 1 INDEXES

Documents Aggregations Schema Explain Plan Indexes Validation

Filter Type a query: { field: 'value' } Reset Find More Options

ADD DATA EXPORT COLLECTION 1 - 7 of 7

```
{
  "_id": ObjectId('641c1afea9395bfa879a1113'),
  "id_juego": "1",
  "unidades": 2
}
```

1 _id: ObjectId('641c1afea9395bfa879a1114') ObjectId
2 id_juego: "5" String
3 unidades: 2 Int32

Document modified. CANCEL UPDATE

```
{
  "_id": ObjectId('641c1afea9395bfa879a1114'),
  "id_juego": "5",
  "unidades": 6
}
```

De la misma manera introducimos los datos en la colección **videojuegos**.

Data Warehouse

Documents proyecto.videojue...

proyecto.videojuegos

8 DOCUMENTS 1 INDEXES

Documents Aggregations Schema Explain Plan Indexes Validation

Filter Type a query: { field: 'value' } Reset Find More Options

ADD DATA EXPORT COLLECTION 1 - 8 of 8


```
{
  "_id": ObjectId('641c42ebbd489b22d1e5d57'),
  "nombre": "Mario y Luigi",
  "consola": "nintendo",
  "id": "1"
}
```


```
{
  "_id": ObjectId('641c42ebbd489b22d1e5d58'),
  "nombre": "The last of us",
  "consola": "play",
  "id": "2"
}
```


```
{
  "_id": ObjectId('641c42ebbd489b22d1e5d59'),
  "nombre": "Sony",
  "consola": "sega",
  "id": "3"
}
```

Conectamos el [Data Federation](#) con la base de datos **proyecto**

Choose Your Data Store to Get Started [View Data Federation Docs](#)

 **Amazon S3**

 **Atlas Cluster**

 **HTTP(S)**

Choose the Atlas clusters that you would like to access in your Federated Database instance.

Provide Namespaces in this project

> ☐ console

> ☐ juegos

> ☒ proyecto

> ☐ prueba_mia

> ☐ sucursales

> ☐ Tienda

Cluster Read Preference >

Back

Cancel

Next

Ya las tenemos en el [Data Lake](#)

Atlas

Cluster: iebs

Access Manager

Billing

All ClustersGet HelpManiya

Data WarehouseData ServicesApp ServicesCharts

DEPLOYMENTDatabaseData LakePREVIEWSERVICESTriggersData APIData FederationSearchSECURITYDatabase AccessNetwork AccessAdvancedNew On AtlasGoto

Add Data Sources

Filter: All data sources

ATLAS CLUSTER

Cluster Store: iebs

juegos.playX

juegos.xboxX

console.nintendoX

Tienda.ProductosX

Tienda.VentasX

proyecto.ventasX

proyecto.videojuegosX

Federated Database Instance: FederatedDatabaseInstance0

Add Database

juegos

prueba

Tienda

proyecto

ventas

proyecto.ventasATLAS CLUSTERX

Drag the dataset to your Federated Database

videojuegos

proyecto.videojuegosATLAS CLUSTERX

Drag the dataset to your Federated Database

Volvemos a [MongoDB Compass](#) para hacer las consultas

The screenshot shows the MongoDB Compass interface. On the left, the 'Data Lake' sidebar shows the 'proyecto' database with collections 'ventas', 'videojuegos', and 'prueba'. The main panel shows the 'Collections' tab with details for 'ventas' and 'videojuegos'. The 'ventas' collection has 7 documents and a storage size of 364.00 B. The 'videojuegos' collection has 8 documents and a storage size of 520.00 B.

Vamos a calcular las unidades vendidas para cada juego:

Para hacerlo lo primero que tenemos que hacer es unir las dos colecciones, la de [ventas](#) y la de [videojuegos](#). Esto se hace con la operación **\$lookup**

The screenshot shows the MongoDB Compass interface with the '\$lookup' aggregation stage selected. The query is:

```
1 {
2   from: 'videojuegos',
3   localField: 'id_juego',
4   foreignField: 'id',
5   as: 'Totales'
6 }
```

The output shows a sample of 7 documents. The first document is:

```
{
  "_id": ObjectId("641c1afea9395bfa879a1113"),
  "id_juego": "1",
  "unidades": 2,
  "Totales": Array
}
```

A red arrow points from the 'Totales' field in the first document to the next screenshot.

Nos lo muestra como un array. Lo convertimos en objeto con **\$unwind**

The screenshot shows the MongoDB Compass interface with the '\$unwind' aggregation stage selected. The query is:

```
1 {
2   path: '$Totales'
3 }
```

The output shows a sample of 7 documents. The first document is:

```
{
  "_id": ObjectId("641c1afea9395bfa879a1113"),
  "id_juego": "1",
  "unidades": 2,
  "Totales": Object
}
```

A red arrow points from the 'Totales' field in the first document to the next screenshot.

Agrupamos por [Consola](#) y [Nombre_Juego](#) y sumamos las [Unidades](#):

The screenshot shows the MongoDB Compass interface with the '\$group' aggregation stage selected. The query is:

```
1 {
2   _id: {
3     Consola: '$Totales.consola',
4     Nombre_Juego: '$Totales.nombre',
5   },
6   Unidades: {
7     $sum: '$unidades'
8   }
9 }
```

The output shows a sample of 5 documents. The first document is:

```
{
  "_id": Object,
  "Consola": "pc",
  "Nombre_Juego": "lol",
  "Unidades": 7
}
```

Sacamos **Consola** y **Nombre_Juego** fuera del **Objeto** con **\$set**:



```
1 {
2   Nombre_Juego: '$_id.Nombre_Juego',
3   Consola: '$_id.Consola',
4 }
```

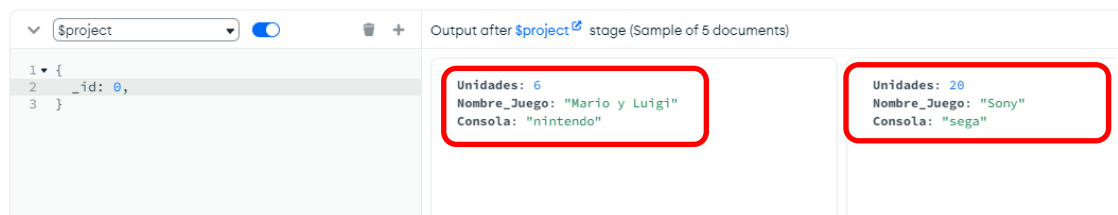
Output after **\$set** stage (Sample of 5 documents)

```
{
  _id: Object,
  Unidades: 7,
  Nombre_Juego: "lol",
  Consola: "pc"
}
```

```
{
  _id: Object,
  Unidades: 5,
  Nombre_Juego: "Forza Horizon",
  Consola: "xbox"
}
```

Quitamos el campo **_id**, para que no se vea con **\$project** y ver solo lo que se pide en la definición:

{ 'Unidades': 100, 'Nombre_Juego': 'Crash Bandicoot', 'Consola': 'Play' }.



```
1 {
2   _id: 0,
3 }
```

Output after **\$project** stage (Sample of 5 documents)

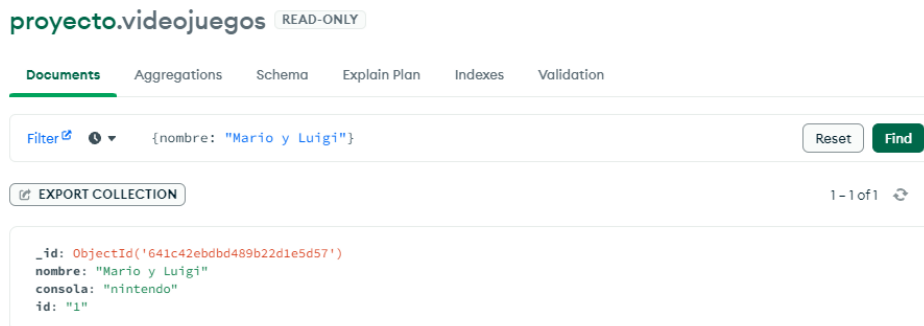
```
{
  Unidades: 6,
  Nombre_Juego: "Mario y Luigi",
  Consola: "nintendo"
}
```

```
{
  Unidades: 20,
  Nombre_Juego: "Sony",
  Consola: "sega"
}
```

Nos muestra que se han vendido 6 unidades del juego **"Mario y Luigi"** para la consola **"Nintendo"**.

Lo comprobamos:

Vemos que el juego con el nombre **"Mario y Luigi"** se ha vendido solo para la consola **"nintendo"**.



proyecto.videojuegos READ-ONLY

Documents Aggregations Schema Explain Plan Indexes Validation

Filter {nombre: "Mario y Luigi"} Reset Find

EXPORT COLLECTION 1 - 1 of 1

```
{
  _id: ObjectId('641c42ebdbd489b22d1e5d57'),
  nombre: "Mario y Luigi",
  consola: "nintendo",
  id: "1"
}
```

Vemos que tiene **id : "1"** y buscamos cuantas unidades se han vendido :



proyecto.ventas READ-ONLY

Documents Aggregations Schema Explain Plan Indexes Validation

Filter {id_juego: "1"} Reset Find

EXPORT COLLECTION 1 - 2 of 2

```
{
  _id: ObjectId('641c1afea9395bfa879a1113'),
  id_juego: "1",
  unidades: 2
}
```

```
{
  _id: ObjectId('641c1bd8a9395bfa879a1119'),
  id_juego: "1",
  unidades: 4
}
```

Y para la otra consulta que se pide en la definición:

Las unidades vendidas totales de cada consola

{'Unidades': 200, 'Consola': 'Xbox' }

En este caso haremos otra consulta, también uniremos las dos colecciones por el campo **id**,

```
1 {
2   from: "videojuegos",
3   localField: "id_juego",
4   foreignField: "id",
5   as: "Totales",
6 }
```

Output after \$lookup stage (Sample of 7 documents)

Document 1:
_id: ObjectId('641c1afea9395bfa879a1113')
id_juego: "1"
unidades: 2
Totales: Array

Document 2:
_id: ObjectId('641c1afea9395bfa879a1114')
id_juego: "5"
unidades: 6
Totales: Array

Convertimos el array en objeto:

```
1 {
2   path: "$Totales",
3 }
```

Output after \$unwind stage (Sample of 7 documents)

Document 1:
_id: ObjectId('641c1afea9395bfa879a1113')
id_juego: "1"
unidades: 2
Totales: Object

Document 2:
_id: ObjectId('641c1afea9395bfa879a1114')
id_juego: "5"
unidades: 6
Totales: Object

Y agruparemos por **consola**.

```
1 {
2   _id: {
3     Consola: "$Totales.consola",
4   },
5   Unidades: {
6     $sum: "$unidades",
7   },
8 }
```

Output after \$group stage (Sample of 5 documents)

Document 1:
_id: Object
Consola: "nintendo"
Unidades: 6

Document 2:
_id: Object
Consola: "xbox"
Unidades: 5

Sacamos el campo **Consola** fuera del objeto:

```
1 {
2   Consola: "$_id.Consola",
3 }
```

Output after \$set stage (Sample of 5 documents)

Document 1:
_id: Object
Consola: "play"
Unidades: 10
Consola: "play"

Document 2:
_id: Object
Consola: "nintendo"
Unidades: 6
Consola: "nintendo"

Nos quedamos solo con los campos **Unidades** y **Consola**

```
1 {
2   _id: 0,
3 }
```

Output after \$project stage (Sample of 5 documents)

Document 1:
Unidades: 20
Consola: "sega"

Document 2:
Unidades: 7
Consola: "pc"

Vemos que para la consola “pc” se han vendido 7 unidades, que una vez se han vendido 6 y otra vez 1.

proyecto.ventas READ-ONLY

Documents Aggregations Schema Explain Plan

Filter {id_juego: "5"}

EXPORT COLLECTION

```
_id: ObjectId('641c1afea9395bfa879a1114')
id_juego: "5"
unidades: 6
```

```
_id: ObjectId('641c1bd8a9395bfa879a1118')
id_juego: "5"
unidades: 1
```

Llegamos hasta aquí buscando el id donde vimos que es “5”, mostrando el campo consola.

proyecto.videojuegos READ-ONLY

Documents Aggregations Schema Explain Plan Indexes Validation

Filter {consola: "pc"} Reset Find

EXPORT COLLECTION 1 - 1 of 1

```
_id: ObjectId('641c42ebdbd489b22d1e5d5b')
nombre: "lol"
consola: "pc"
id: "5"
```