

Explorando o Compass

Índice

- 01 Ações iniciais
- 02 Guia Documents
- 03 Guia Aggregation e Pipelines
- 04 Guia Schema
- 05 Guia Explain
- 06 Guia Indexes
- 07 Guia Validation



01

Ações iniciais

Entrando na ferramenta

Ao entrar na ferramenta, o painel exibirá os dados de conexão, os bancos de dados do sistema e os que já foram criados anteriormente, utilizando o Shell.

Para recolher os dados de conexão, basta clicar no ícone seta, indicado na imagem.



02

Guia Documents

Visualizando documentos de uma coleção

Clique no ícone seta ao lado do Banco de Dados desejado (no exemplo, DBII).

Em seguida, selecione uma coleção para visualizar os documentos.

The screenshot displays the MongoDB Compass interface. On the left sidebar, under the 'Local' section, the 'DBII' database is selected, and the 'contato' collection is highlighted with a yellow box and a yellow arrow. The main panel shows the 'DBII.contato Documents' view. It includes a filter bar with the filter '{ field: 'value' }', a 'FIND' button, and a 'RESET' button. Below the filter bar, there are tabs for 'Documents', 'Aggregations', 'Schema', 'Explain Plan', 'Indexes', and 'Validation'. The 'Documents' tab is active, showing a list of documents. The first document is:

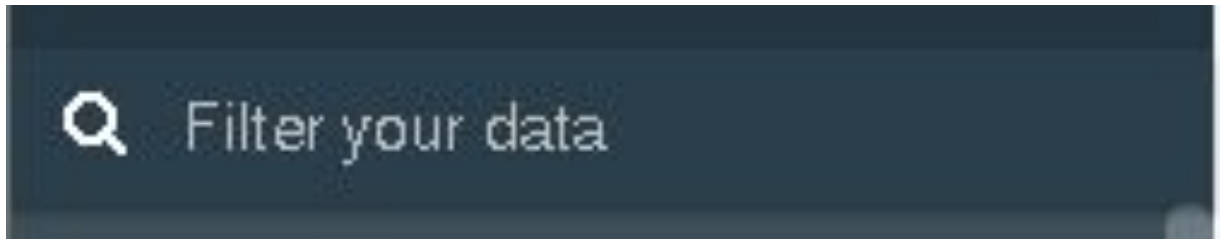
```
{
  "_id": 1,
  "nome": "Ana Luisa",
  "idade": 22,
  "cidade": "Campo Grande",
  "estado": "MS",
  "telefone": "(67) 3032-2233"
}
```

The second document is:

```
{
  "_id": "002",
  "nome": "Luis Claudio",
  "idade": "31",
  "cidade": "Buzios",
  "estado": "RJ",
  "telefone": "(21) 2019-3294"
}
```

Localizando um banco de dados

Para localizar um banco de dados mais rapidamente, digite o nome do banco no campo **Filter your data**.



No painel da figura abaixo, visualize o nome do banco de dados e a coleção selecionada, a quantidade de documentos e índices.



Guia Documents

Utilize o filtro para localizar qualquer termo pertencente à coleção.

Você deve digitar as chaves, o nome do campo e o valor: ex: `{estado: "DF"}`. Em seguida, clique em **Find** para localizar.

The screenshot shows the DBII.contato MongoDB interface. At the top, it displays database statistics: 6 documents, 20.5KB storage size, 121B avg size, 1 index, 36.9KB total size, and 36.9KB avg index size. Below this is a navigation bar with tabs for Documents, Aggregations, Schema, Explain Plan, Indexes, and Validation. The 'Documents' tab is active. A search bar contains the filter `{estado: "DF"}`. To the right of the search bar are buttons for 'FILTER', 'OPTIONS', 'FIND', 'RESET', a refresh icon, and a menu icon. Below the search bar is a toolbar with an 'ADD DATA' button, an upload icon, a 'VIEW' button, and three view mode icons (list, JSON, grid). The status bar indicates 'Displaying documents 1 - 1 of 1' with navigation arrows and a 'REFRESH' button. The document being displayed is:

```
{
  "_id": "003",
  "nome": "Carlos Alberto",
  "idade": "26",
  "cidade": "Brasília",
  "estado": "DF",
  "telefone": "(61) 3233-3487"
}
```

Botão OPTIONS para filtrar o documento

Clicando em Options, temos ainda, a opção de filtrar nosso documento, escolhendo as colunas que queremos que sejam exibidas e ordenar nosso resultado.



The image shows a MongoDB query builder interface with several sections for constructing a query:

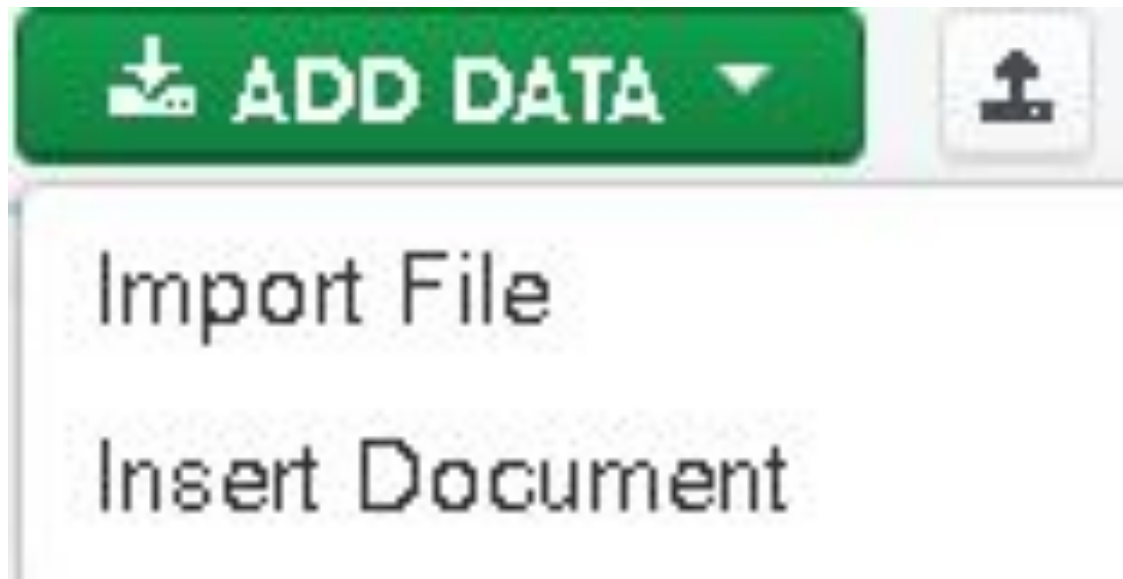
- FILTER**: { field: 'value' }
- PROJECT**: { field: 0 }
- SORT**: { field: -1 } or [['field', -1]]
- COLLATION**: { locale: 'simple' }
- MAX TIME MS**: 60000
- SKIP**: 0
- LIMIT**: 0
- OPTIONS**: A dropdown menu button in the top right corner, highlighted with a blue box and a yellow arrow pointing to it.

Criando um novo documento

Para criar um novo documento, clique em **ADD Data** e escolha uma das opções:

- **Import File** - para importar um arquivo.
- **Insert Document** - para criar um documento na coleção.

Ao selecionar a opção **Insert Document**, você pode escolher a forma como quer incluir o documento: **modo JSON** ou modo **Editor campo-a-campo**.



Inserindo documentos: modo JSON

A imagem demonstra a inserção de um documento no modo Json.

Insert to Collection DBII.contato

VIEW  

```
1 ▾ /**
2   * Paste one or more documents here
3   */
4 ▾ {
5 ▾   "_id": {
6     "$oid": "61face35ab8fed2d8b275564"
7   }
8 }
```

Inserindo documentos: modo Editor campo-a-campo

A imagem demonstra a inserção de um documento no modo editor campo-a-campo.

Insert to Collection DBII.contato

VIEW { } ≡

1

`_id: ObjectId("61face35ab8fed2d8b275564")`

ObjectId

Cancel

Insert

03

Guia Aggregation e Pipelines

Aggregation

Utilizamos a guia

Aggregation para criarmos um **Pipeline de Agregação**.

Adicionamos estágios para utilizarmos funções ou filtros na nossa coleção, conforme a necessidade, como selecionar os campos que queremos que sejam exibidos, contar documentos, agrupar os dados, concatenar campos e muito mais.

The screenshot shows the MongoDB Compass interface. At the top, the database name 'DBII.contato' is displayed. To the right, statistics are shown: 6 DOCUMENTS, 20.5KB STORAGE SIZE, 121B AVG. SIZE, 1 INDEXES, 36.9KB TOTAL SIZE, and 36.9KB AVG. SIZE. Below this, a navigation bar includes 'Documents', 'Aggregations' (which is highlighted with a green underline), 'Schema', 'Explain Plan', 'Indexes', and 'Validation'. The main interface has a top toolbar with icons for undo, redo, save, and a dropdown menu. Below the toolbar, there's a section for '6 Documents in the Collection' with a refresh icon and a 'Preview of Documents in the Collection' button. The central area is divided into two panes. The left pane contains the text: 'Select an operator to construct expressions used in the aggregation pipeline stages. [Learn more](#)'. The right pane displays a JSON document:

```
{
  "_id": 1,
  "nome": "Ana Luisa",
  "idade": 22,
  "cidade": "Campo Grande",
  "estado": "MS",
  "telefone": "(67) 3032-2233"
}
```

 At the bottom, there's a status bar with a 'Select...' dropdown, a toggle switch, and a 'A sample of the aggregated results from this stage will be shown below' message.

Aggregation

Exemplo de implementação por estágios:

Utilizando a coleção **contato**, vamos selecionar o estágio **\$project**, onde definiremos os campos que queremos exibir. Observe a imagem.



Aggregation

Nosso primeiro estágio foi definido. Seleccionamos apenas os campos **nome**, **idade** e **estado** para serem exibidos.


Automaticamente, do lado direito, já mostra como ficará a visualização.

Vamos **adicionar mais um estágio**, clicando no botão **ADD STAGE**:



Aggregation

Vamos definir, agora, um filtro para nossa visualização. Vamos listar apenas os contatos com **idade maior que 30 anos**. Para isso, utilizaremos a função **\$match**:



The screenshot shows a MongoDB aggregation pipeline editor. The left pane displays a query in MQL, and the right pane shows the output after the \$match stage.

Query Editor:

- Line 1: `1 ▾ /**`
- Line 2: `2 * query: The query in MQL.`
- Line 3: `3 */`
- Line 4: `4 ▾ {`
- Line 5: `5 idade: { $gt: 30 }`
- Line 6: `6 }`

Output: Output after `$match` stage ⓘ (Sample of 3 documents)

```
nome: "Luís claudio"
idade: 31
estado: "RJ"
```

Aggregation

Podemos, ainda, ordenar nossa lista. Vamos adicionar mais um estágio, classificando a coluna **nome** em **ordem crescente**. Em seguida, você poderá observar que os documentos estarão sendo exibidos por ordem alfabética.



The screenshot shows a MongoDB query editor interface. On the left, a JSON query is defined with a `$sort` stage. The `nome` field is set to sort in ascending order (1). On the right, the output of the `$sort` stage is displayed, showing a sample of 3 documents. The first document shown is:

```
{
  "nome": "José Roberto",
  "idade": 35,
  "estado": "SP"
}
```

A cada operação que quisermos adicionar à nossa consulta, vamos criando um estágio.

Aggregation

Agora vamos agrupar os contatos por estado. Neste exemplo, exibimos a quantidade de contatos existentes, maior de 30 anos, em cada estado.

The screenshot shows a MongoDB aggregation pipeline editor. The top toolbar includes a dropdown menu, a folder icon, a plus icon, a dropdown menu, a 'COLLATION' button, a 'Untitled- Modified' text field, a green 'SAVE' button, and a document icon. On the right, there are two toggle switches: 'SAMPLE MODE' and 'AUTO PREVIEW', both of which are turned on.

The main editor area has a dropdown menu set to '\$group' and a green toggle switch. Below this, the aggregation pipeline is defined in a code editor:

```
1 ▾ /**  
2  * _id: The id of the group.  
3  * fieldN: The first field name.  
4  */  
5 ▾ {  
6   _id: "$estado",  
7   Total: {  
8     $sum: 1  
9   }  
10 }
```

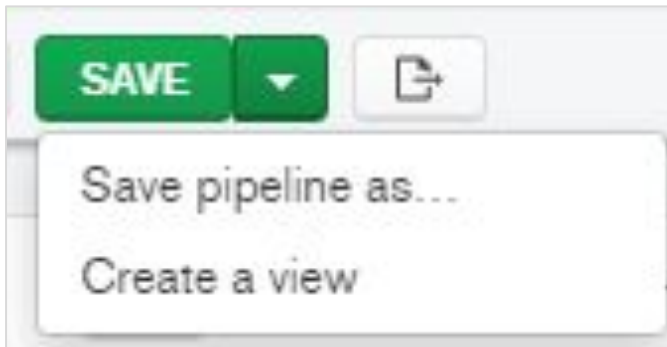
Below the code editor, the output of the aggregation is displayed. The text 'Output after \$group stage ⓘ (Sample of 3 documents)' is shown. Two sample output documents are displayed in separate boxes:

```
_id: "MG"  
Total: 1
```

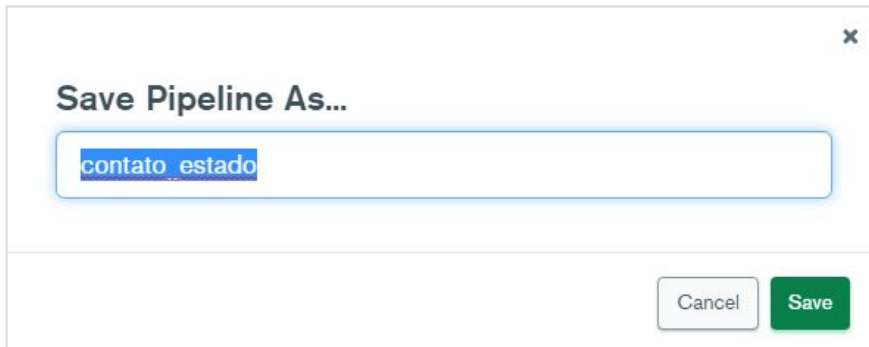
```
_id: "RJ"  
Total: 1
```

Salvando um Pipeline

Agora que já utilizamos todos os filtros e funções que entendemos necessários à nossa consulta, temos a opção de salvar o pipeline ou criar uma view. Observe as imagens.



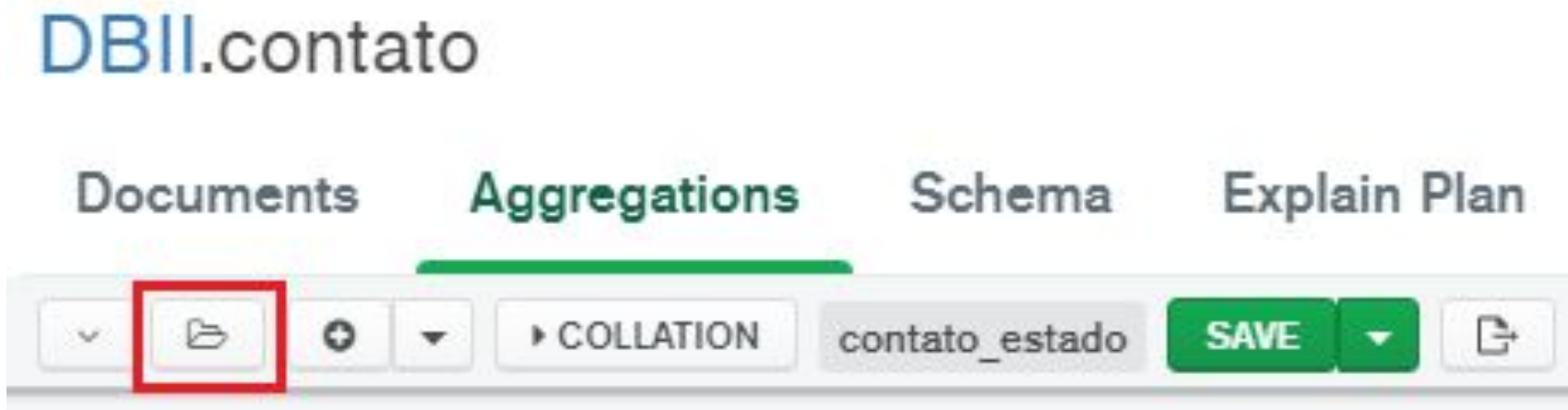
Selecionando a opção **Save pipeline as...** podemos salvar nossa consulta para usar em outro momento.



Escolha um nome sugestivo e clique no botão **Save**.

Abrindo um Pipeline

Para abrir o pipeline que você criou, clique no ícone pasta, localizado abaixo das abas, no canto superior esquerdo da tela.



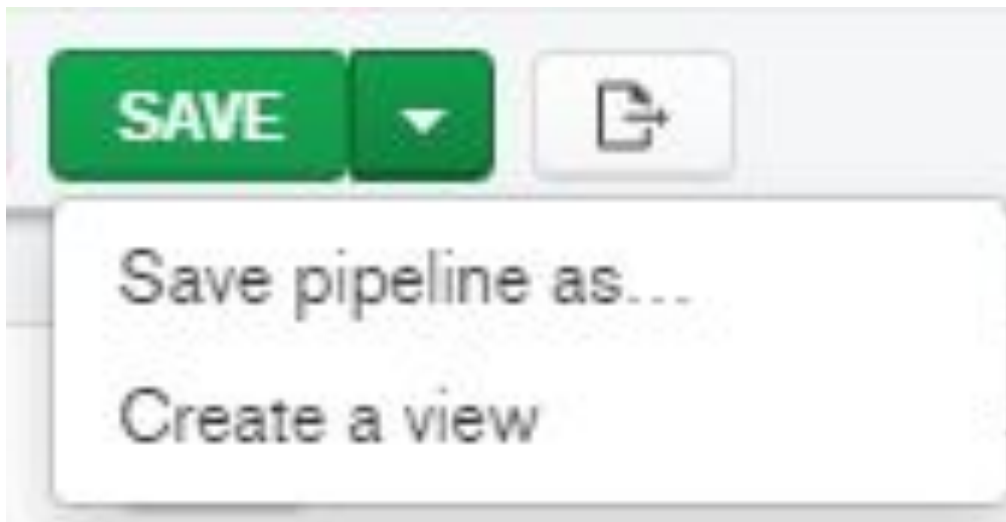
Abrindo um Pipeline

Aponte o mouse para o pipeline desejado e clique em OPEN. Agora, role a tela para ver os estágios que você definiu.



Criando uma View

A opção **Create a view** permite criarmos uma visualização a partir do resultado do pipeline, mas não salva o pipeline propriamente dito.



Criando uma View

Insira um nome para sua view e clique em **Create**. A view dos resultados do pipeline será criada no mesmo banco de dados em que o pipeline foi criado.



Create a View

Enter a View Name

Cancel

Create

Visualizando os resultados

Depois de criada a view dos resultados do pipeline, observe no banco de dados utilizado que a view já é exibida. Quando precisar, basta clicar sobre ela para o resultado ser exibido.



Exportando seu pipeline

O Compass ainda permite que o pipeline seja exportado para uma linguagem de programação, para ser utilizado no seu projeto. Escolha entre Java, Node, C# ou Python e acelere o seu desenvolvimento.

Você pode, também, simplesmente copiar seu código para executar no MongoDB Shell.



Exportando seu pipeline

Para copiar o código do pipeline, clique no ícone **Export To Language**.

Clique no ícone Copiar, feche a janela e cole no terminal.

Para executar, digite antes do código copiado: **db.collection.aggregate(** , fechando o parêntese ao final do código).

Lembre-se: **collection** é o nome da coleção que você estiver trabalhando.

Export Pipeline To Language

My Pipeline:

```
1 [{
2   $group: {
3     _id: '$genero',
4     musicas: {
5       $sum: 1
6     },
7     titulo: {
8       $push: '$musica'
9     }
10  }
11 }]
```



Exportando seu pipeline

Para exportar o código do pipeline, clique no ícone **Export To Language**.

Selecione a linguagem desejada, clique em copiar. O código ficará na área de transferência. Abra seu projeto e cole no local desejado.

Opcional: inclua instruções de importação e sintaxe do driver, se desejar.

Marque a opção **Include Import Statements** para incluir as instruções de importação necessárias para a linguagem de programação selecionada.

Marque a opção **Include Driver Syntax** para incluir o código do aplicativo para a linguagem de programação selecionada.

Export Pipeline To: PYTHON 3

```
1 [
2   {
3     '$group': {
4       '_id': '$genero',
5       'musicas': {
6         '$sum': 1
7       },
8       'titulo': {
9         '$push': '$musica'
10      }
11    }
12  }
13 ]
```

☐ Include Import Statements

☐ Include Driver Syntax

CLOSE

04

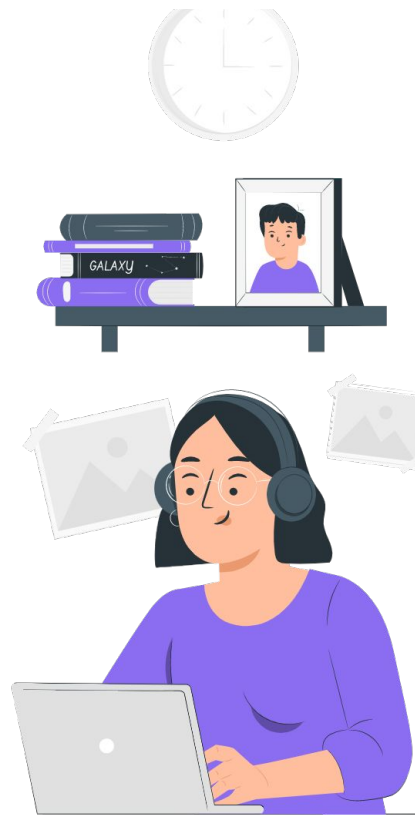
Guia Schema

Guia Schema

A guia **Schema** fornece uma visão geral do **tipo de dados** e da **forma dos campos** em uma **coleção específica**.

Para essa análise, não é necessário exibir todos os dados da coleção. O Compass trabalha com **amostras**.

No nosso exemplo, analisaremos a coleção contato. Observe-a na imagem da tela a seguir.



FILTER

{ field: 'value' }

► OPTIONS

ANALYZE

RESET



This report is based on a sample of 6 documents. ⓘ

_id

string

double



- 004
- 006
- 005

cidade

string



- Campo Grande

Coleção contato

Observe que a análise traz o nome do campo, o tipo de dado e o conteúdo.

No campo **_id**, nos é informado que tempos alguns campos string e outros double.

FILTER

{ field: 'value' }

► OPTIONS

FIND

RESET



ADD DATA ▼



VIEW



Displaying documents 1 - 6 of 6



REFRESH

```
1  _id: 1
2  nome: "Ana Luisa"
3  idade: 22
4  cidade: "Campo Grande"
5  estado: "MS"
6  telefone: "(67) 3032-2233"
```

Guia Documents

Podemos, então, facilmente, na guia Documents, alterar o tipo de dado do campo id, selecionando o tipo de dado correto.

Double
String
Int32
String
String
String

CANCEL

UPDATE

```
1  _id: "002"
2  nome: "Luís Claudio"
3  idade: 31
4  cidade: "Buzios"
5  estado: "RJ"
6  telefone: "(21) 3333-3333"
```

string
string
Int32
string
string
string

05

Guia Explain

Guia Explain

Na guia **Explain Plain** (Planos de Execução), você entende como sua consulta foi executada. E, também, se você pode melhorar esse comportamento.

No exemplo ao lado, listamos os dados da coleção contato. Observe na imagem.

DBII.contato

Documents Aggregations Schema **Explain Plan** Indexes Validation

FILTER { field: 'value' } OPTIONS EXPLAIN RESET ↺ ...

VIEW DETAILS AS VISUAL TREE RAW JSON

Query Performance Summary

| | |
|------------------------|--------------------------------------|
| Documents Returned: 6 | Actual Query Execution Time (ms): 20 |
| Index Keys Examined: 0 | Sorted in Memory: no |
| Documents Examined: 6 | ⚠ No index available for this query. |

COLLSCAN

nReturned: 6

O Explain nos informa que a consulta retornou 6 documentos, em 20 ms (milissegundos), e que não foi utilizado índice.

06

Guia Indexes

Guia Indexes

A guia Indexes exibe os índices que foram criados na coleção selecionada e quais foram usados.

Essa é uma informação importante, pois pode sugerir alterações na sua query ou no índice.

Neste caso, temos um índice chamado `_id_`, do tipo unique, com 36.9 kb de tamanho, utilizado por 23 vezes.

DBII.contato

Documents Aggregations Schema Explain Plan **Indexes** Validation

CREATE INDEX

| Name and Definition ▲ | Type | Size | Usage | Properties |
|--|-----------|------------------------|-----------------------------|------------|
| <code>_id_</code> <code>_id...</code> | REGULAR ⓘ | 36.9 KB <div></div> | 23 since Wed Feb 02 2022 | UNIQUE ⓘ |

**Nota: Uma única coleção não pode ter mais de 64 índices.
Não pode haver mais de 32 campos em um índice composto.**

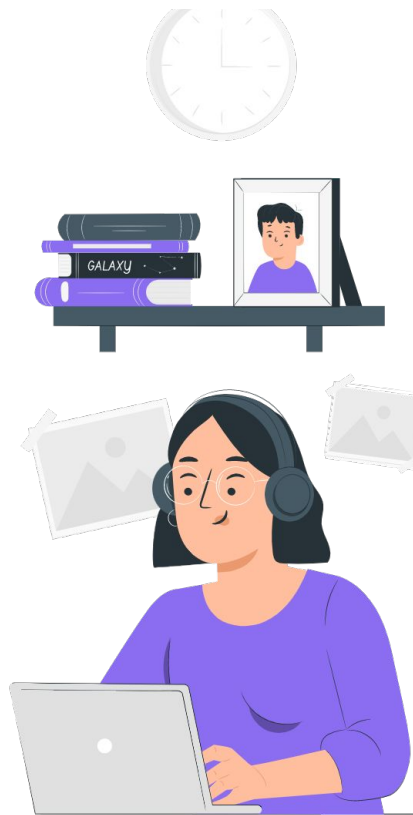
07

Guia Validation

Guia Validation

A guia **Validation** serve para definir **regras de validação** para o seu **schema**.

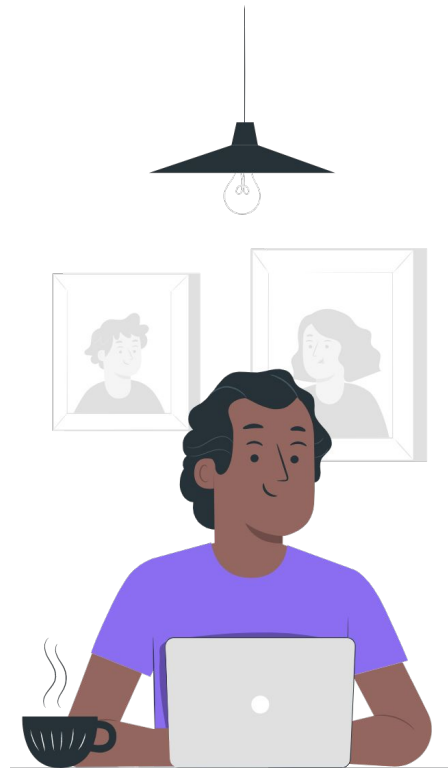
A validação de esquema garante que todos os documentos em uma coleção sigam um conjunto definido de regras. Como, por exemplo, estar em conformidade com uma forma específica ou permitir apenas um intervalo especificado de valores em campos.



Guia Validation

O editor de validação suporta validação de **esquema JSON** e validação com **expressões de consulta**.

Conforme você edita, o Compass é atualizado em tempo real para exibir um documento de sua coleção que passa na validação e um documento que falha.



Guia Validation

Observe o exemplo de validação ao lado.

Neste exemplo, definimos como regra que o campo `_id` deve ser do **tipo double**.

Validation Action ⓘ

ERROR ▾

Validation Level ⓘ

STRICT ▾

```
1 {  
2   $jsonSchema: {  
3     required: [  
4       '_id'  
5     ],  
6     properties: {  
7       _id: {  
8         bsonType: 'double',  
9         description: 'Inclua um valor do tipo double'  
10      }  
11    }  
12  }  
13 }
```

Guia Validation

No exemplo da tela anterior, definimos como regra que o campo `_id` é do **tipo double**. Vimos, na guia Schema, que temos dois tipos de dados no campo `_id`: **double** e **string**. Logo, nosso documento de `_id = 1` foram validados, pois apenas ele possuía o `_id` do tipo double. Os outros não estão em conformidade com a regra de validação.



Sample Document That Passed Validation

```
_id: 1
nome: "Ana Luísa"
idade: 22
cidade: "Campo Grande"
estado: "MS"
telefone: "(67) 3032-2233"
```



Sample Document That Failed Validation

```
_id: "002"
nome: "Luís Claudio"
idade: 31
cidade: "Buzios"
estado: "RJ"
telefone: "(21) 2019-3294"
```

Conclusões

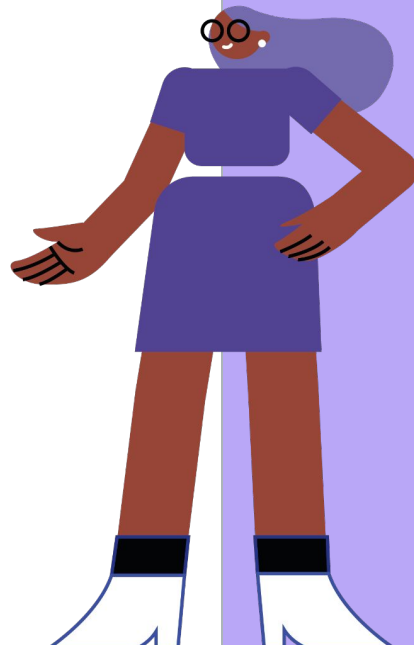
o MongoDB Compass é um grande aliado do desenvolvedor, visto todas as facilidades e possibilidades que essa ferramenta disponibiliza.

Vimos aqui que o Compass permite criar um banco de dados, coleções, documentos e importar dados. Seja em formato Json, seja em CSV.

Podemos, ainda, agregar filtros e funções de forma simplificada, na guia Aggregate. E, também, salvar o pipeline, exportar para uma linguagem de programação ou criar uma view para uso posterior.

O Compass também nos permite visualizar dados importantes, como Schema, índice e ainda criar regras de validação para nosso banco de dados.

Muita informação? Então reveja este tópico e faça o questionário. Estaremos juntos nos próximos desafios!



Muito obrigado!