ENTREGA FINAL BANCO DE DADOS FULLTURE 2024

JANEIRO 2024

FULLTURE
Criado por: Gustavo Antonio Pereira

A EVOLUÇÃO DOS BANCOS DE DADOS.

QUAL SUA IMPORTÂNCIA NO MUNDO CORPORATIVO?

Quando começou toda essa evolução dos bancos de dados? Começou lá atrás, nos anos 60 e 70, quando uma empresa (IBM) percebeu que estava muito custoso contratar várias pessoas para ficar armazenando dados e organizando arquivos. Muitas pesquisas foram realizadas durante esse período. Uma das pesquisas publicadas pelo pesquisador da IBM, Ted Codd, apresentou o primeiro artigo sobre bancos de dados relacionais. Esse artigo discutia o uso de cálculo e álgebra relacional para permitir que usuários não técnicos armazenassem e recuperassem grande quantidade de informações. com os bancos de dados tradicionais (relacional). As pesquisas evoluíram e foi criada também a linguagem SQL (Structured Query Language - Linguagem de Consulta Estruturada). Essa linguagem é utilizada para descrever as operações em um banco de dados e é a bastante utilizada atualmente. Era só guardar e pegar dados organizadinhos.

Mas, à medida que os dados foram crescendo, e novas tecnologias aparecendo, surgiram os bancos de dados NoSQL (não relacional). O termo banco de dados não relacional surgiu pela primeira vez em 1998 por Carlo Strozzi, quando falou sobre um banco de dados não relacionais de código aberto. No ano de 2006 o termo foi novamente citado em um artigo de armazenamento de dados da empresa Google. E em 2009 o termo NoSQL foi mencionado durante um evento da Rackspace sobre banco de dados open source. Ele possibilita a flexibilidade na hora de armazenar os dados, já que não se limita a tabelas com linhas e colunas. NoSQL tem um modelo de armazenamento otimizado, que é adaptável para o requisito específico de cada dado, por exemplo: possibilita que os dados sejam armazenados como chave/valor simples; documento no formato JSON, BSON, ou até mesmo em forma de gráfico, composto de bordas e vértices. Eles foram tipo uma revolução, lidando bem com dados bagunçados e se adaptando fácil.

Depois, a nuvem entrou em cena. Os bancos de dados na nuvem são tipo o paraíso pra empresas. Facilitam a escalada, o gerenciamento e você nem precisa se preocupar com o hardware.

Temos os bancos de dados distribuídos e de big data. Com eles, dá pra lidar com um volume gigantesco de dados e análises malucas. São essenciais pra coisas como análise de dados grande, como por exemplo Facebook ou google analytics, e as inteligências artificiais como nosso amigo Chat GPT.

Sem esquecer dos bancos de dados in-memory. São rápidos pra caramba, guardando tudo na memória principal. São particularmente adequados para aplicativos que exigem alto desempenho e tempo de resposta rápido. Por exemplo, aplicativos de comércio eletrônico, como lojas online, são usados para guardar temporariamente informações sobre os produtos que têm no estoque, as compras que as pessoas fazem e os perfis dos clientes. Isso é tipo quando você pesquisa algo e o site mostra logo de cara se tem ou não o produto, e quando você compra, a informação é atualizada rapidinho, então você não perde tempo esperando. É assim que garantem uma compra rápida e sem problemas pra gente.

Sabe quando a gente se pergunta por que a empresa, as corporações precisam de um banco de dados? É simples: ele ajuda a guardar todas as informações do nosso site ou sistema interno. Hoje em dia, os bancos de dados são essenciais para a empresa, como a espinha dorsal mesmo, sabia? Eles guardam um montão de dados importantes, quem são nossos clientes, o que eles compram e como a empresa tá se saindo financeiramente. Esses registros são super importantes pra gente entender como as coisas tão indo e pra planejar o que fazer a seguir.

Eles são mega importantes pra gente porque nos ajudam a tomar decisões inteligentes. Tipo, a gente pode olhar os dados e descobrir quais produtos tão vendendo mais, ou entender melhor nossos clientes pra oferecer um atendimento mais bacana.

Com um banco de dados, a gente consegue fazer tudo de forma mais eficiente. Quando alguém precisa de alguma informação, é só procurar lá, que tá tudo guardadinho. Investir em um banco de dados não só ajuda a equipe a ser mais produtiva, mas também melhora a comunicação entre os times e os resultados que a gente alcança. Além disso, eles ajudam a gente a ser mais eficientes. A gente pode ver onde tá rolando algum problema na produção ou onde a gente pode cortar gastos. E isso é vital pra gente crescer e continuar inovando. Sem falar na segurança, né? Com todos esses dados guardados, nós precisamos ter certeza de que ninguém vai meter o nariz onde não deve (kkkk)

O PAPEL DOS SGBD’s, SUAS PRINCIPAIS CARACTERÍSTICAS E SUA ESTRUTURA.

O que é o Teorema CAP e como ele é aplicado?

Sabe aquele programa de computador que ajuda a guardar e organizar um monte de informações da empresa? É isso que a gente chama de Sistema Gerenciador de Banco de Dados (SGBD). Ele é como um armário enorme onde a gente guarda tudo que é importante pro negócio.

Por exemplo, se a gente tem uma loja, precisa anotar informações sobre os produtos, clientes e vendas, né? Em vez de ficar escrevendo tudo em papel ou planilhas, a gente usa o SGBD. Ele é tipo um organizador que deixa tudo arrumadinho e fácil de encontrar.

Os SGBDs têm várias partes importantes:

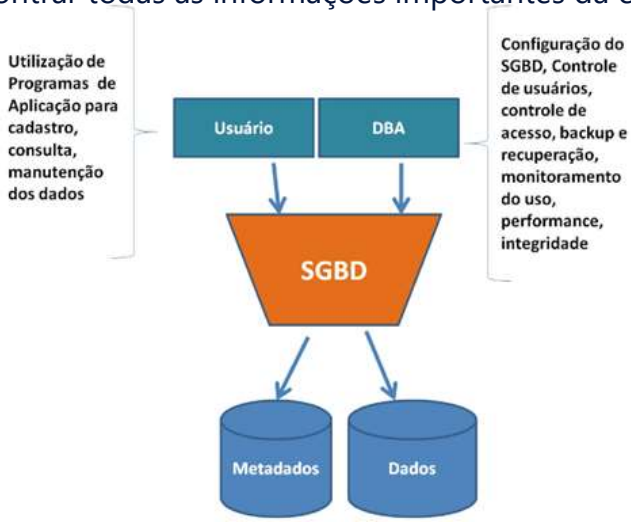
O primeiro é onde a gente guarda todos os dados da empresa, tipo os nomes dos clientes e o preço dos produtos ***(Armazenamento dos Dados***).

Depois, tem uma parte que controla quem pode acessar essas informações. É como se fosse uma chave que só as pessoas autorizadas têm e pode abrir aquela gaveta do guarda roupa ***(Controle de Acesso***).

Também tem uma função de encontrar informações específicas quando a gente precisa. Por exemplo, se a gente quer ver todas as vendas do mês passado, é só pedir pro SGBD buscar esses dados pra gente ***(Recuperação dos Dados***).

E por último, mas não menos importante, ele também protege nossas informações de qualquer perda ou roubo, como se fosse um cofre digital ***(Proteção dos Dados***).

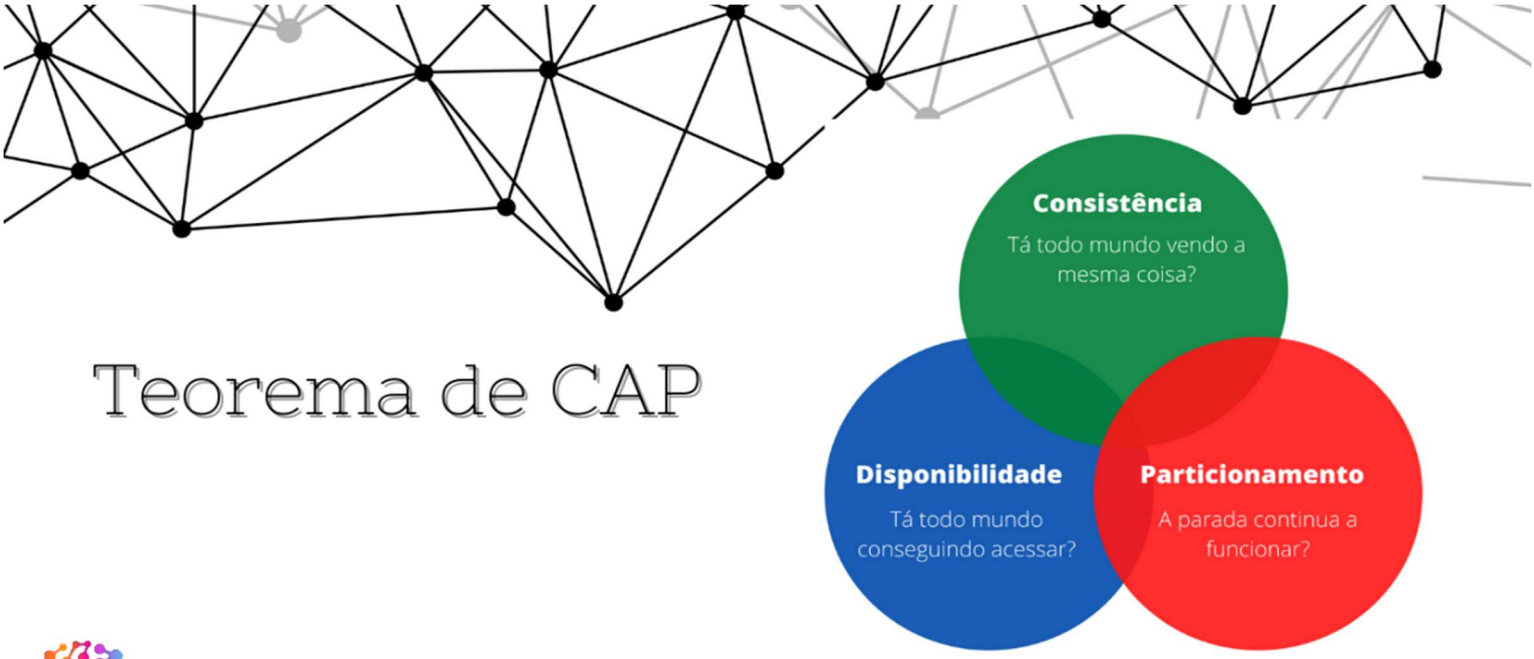
Resumindo, SGBD’s são o conjunto de programas de computador (softwares) por exemplo: ORACLE, MySQL, SQL SERVER, mongoDB, cassandra, super eficientes que ajudam a gente a guardar, proteger e encontrar todas as informações importantes da empresa de forma fácil e segura.



O teorema CAP explicado

O cientista da computação Eric Brewer apresentou o teorema CAP em 1998. Mais tarde, Gilbert e Lynch publicaram uma prova formal do teorema de Brewer sob o nome conjectura de Brewer

CAP significa: Consistency, Availability, Partition tolerance. O teorema afirma que um sistema distribuído não pode garantir todos os três (consistência, disponibilidade e tolerância de partição) o tempo todo. Quando as coisas dão errado, precisamos decidir sobre uma compensação e priorizar no máximo duas características de sistemas distribuídos para manter.



dataeasy

O que é consistência - C?

Um sistema consistente sempre retorna as mesmas informações, independentemente de qual nó consultamos. Imagine o seguinte cenário: Aninha envia dinheiro para sua conta familiar das Filipinas e seu irmão Joãozinho retira dinheiro da mesma conta familiar na Holanda. Ambos os usuários enviam operações de gravação por meio de seu aplicativo financeiro. A operação de Ana foi uma escrita de depósito e a operação de João foi uma escrita de retirada. O aplicativo financeiro usa um banco de dados distribuído, de modo que as gravações foram processadas em locais diferentes - uma vez pelo nó das Filipinas e uma vez pelo servidor holandês. Quando Ana e João verificam o saldo de suas contas para ver se transferiram dinheiro com sucesso, eles desejam ver o estado máximo após as operações de gravação mais recentes, eles não se importam se os "nós" estão entre os globos e precisam sincronizar informações

entre eles. Um sistema consistente mostrará as mesmas informações (a gravação mais recente para a leitura mais recente), independentemente de quais "nós" processam essas informações primeiro.

O que é disponibilidade - A?

Um sistema disponível fornece a cada solicitação de leitura ou gravação uma resposta apropriada (sem erro). Os sistemas de alta disponibilidade alcançam capacidade de resposta por meio da replicação. Eles duplicam dados em vários nós para estarem disponíveis para receber solicitações de leitura ou gravação.

Observe que um sistema disponível não oferece garantias de consistência - ele garante que as solicitações de gravação e leitura serão processadas, mas não que a gravação mais recente será observada.

O que é tolerância de partição - P?

A tolerância de partição se refere à capacidade de um sistema distribuído de funcionar normalmente no momento de falhas de rede.

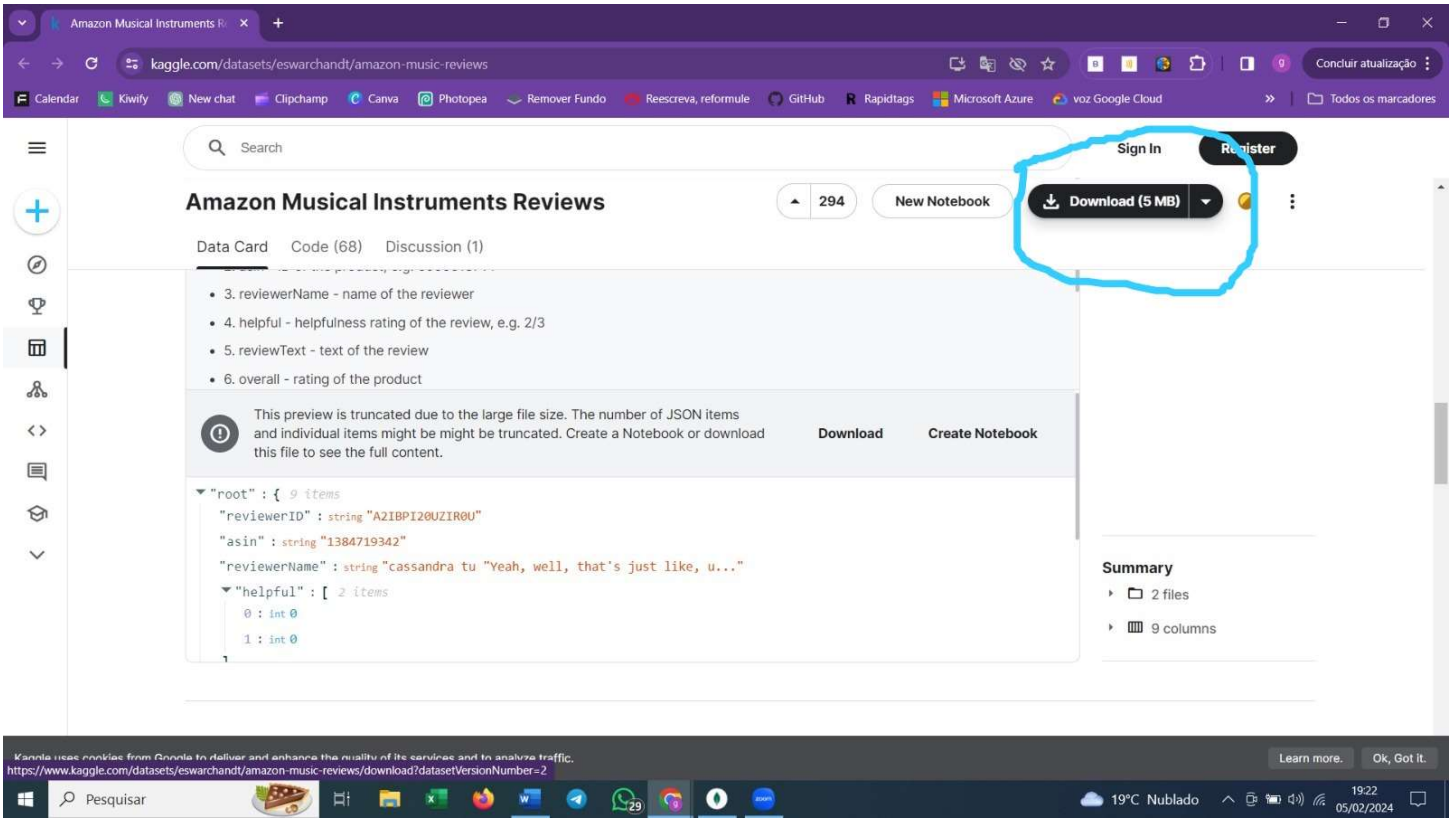
Uma partição de rede é o termo técnico de uma falha de rede - toda a rede é particionada em sub-redes porque um ou dois "nós" perderam a comunicação entre eles. A garantia de tolerância de partição promete que os usuários que enviam solicitações de leitura ou gravação para o sistema não será afetados pelas partições de rede. De um observador externo, o sistema se comportará como se nenhum nó tivesse falhado.

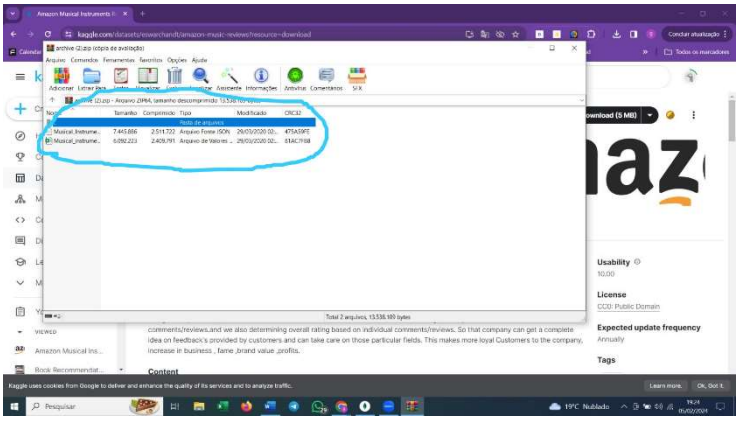
Resumindo, o teorema CAP nos faz escolher entre ter um sistema que sempre está disponível para consulta, mas que pode ser inconsistente (AP), ou um sistema que sempre retorna dados consistentes, mas que pode ficar indisponível em alguns momentos (CP). Embora os sistemas distribuídos geralmente funcionem bem, o aumento do volume e da velocidade dos dados nos faz mais conscientes das compensações que precisamos fazer quando ocorrem erros. Isso levou à popularização de bancos de dados NoSQL, como Cassandra (que prioriza a disponibilidade) ou MongoDB (que prioriza a consistência). Qual escolher depende dos objetivos comerciais de cada empresa: você perderia mais receita se seu sistema ficasse indisponível ou se retornasse dados imprecisos?

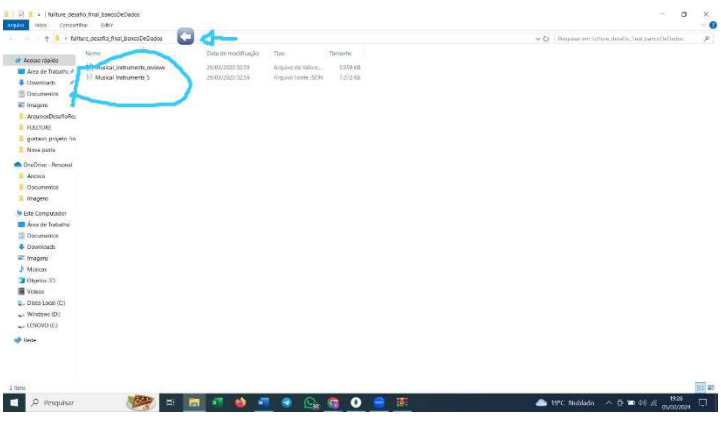
MongoDB PRIMEIROS PASSOS.

Download da dataSet Amazon Musical Instruments Reviews!

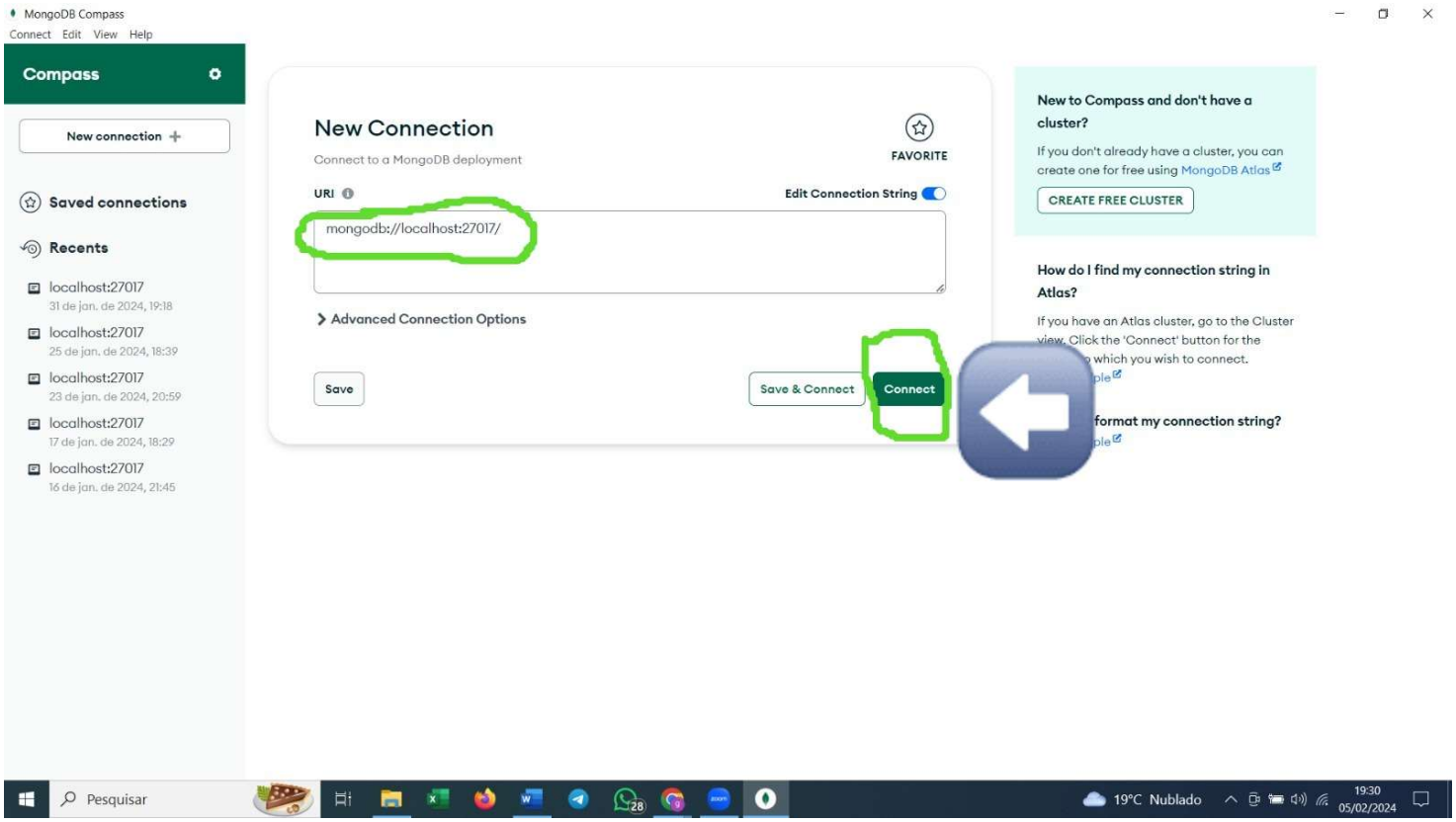
➢ Iniciando com download da dataset. (segue img a baixo)



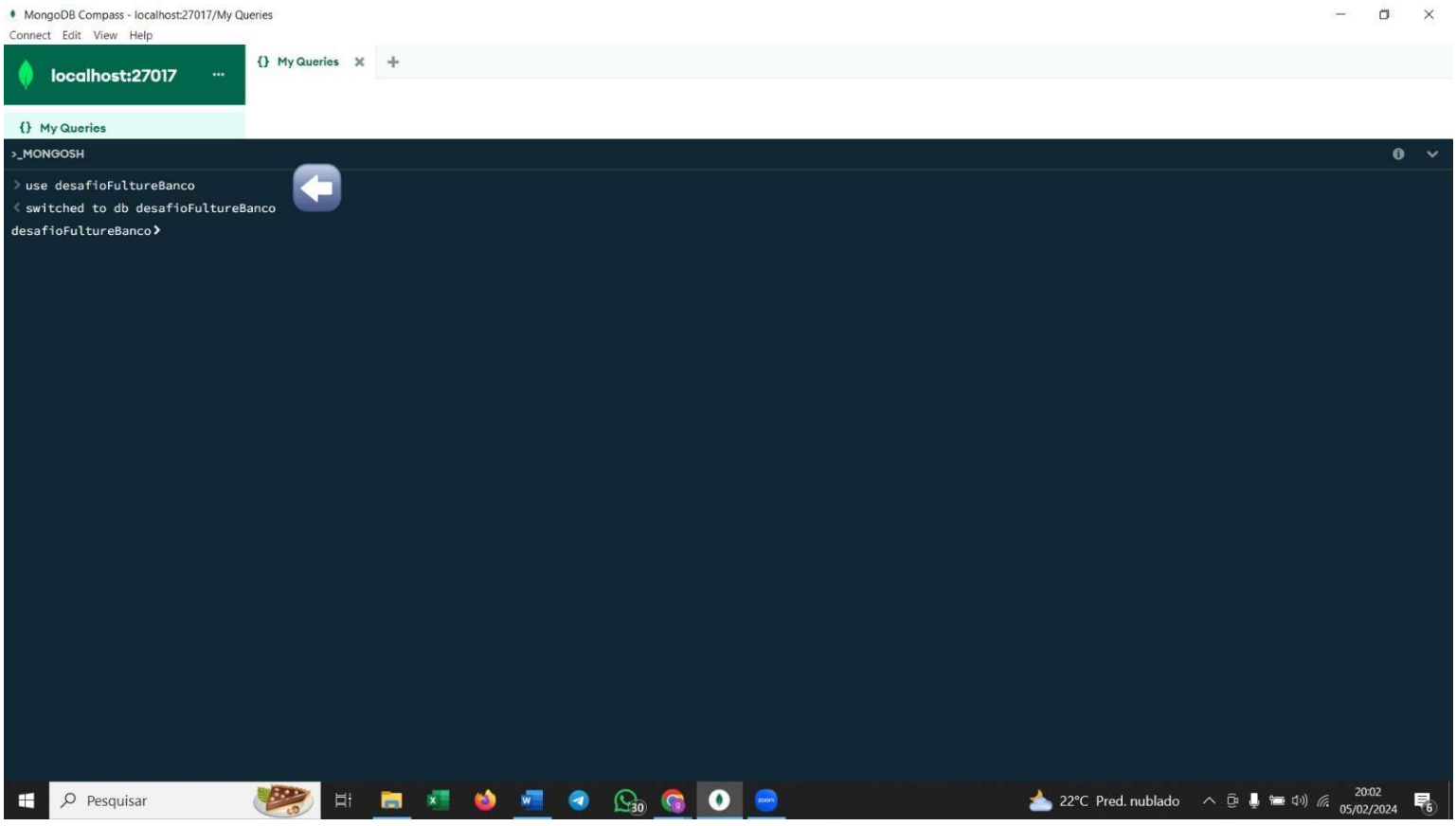




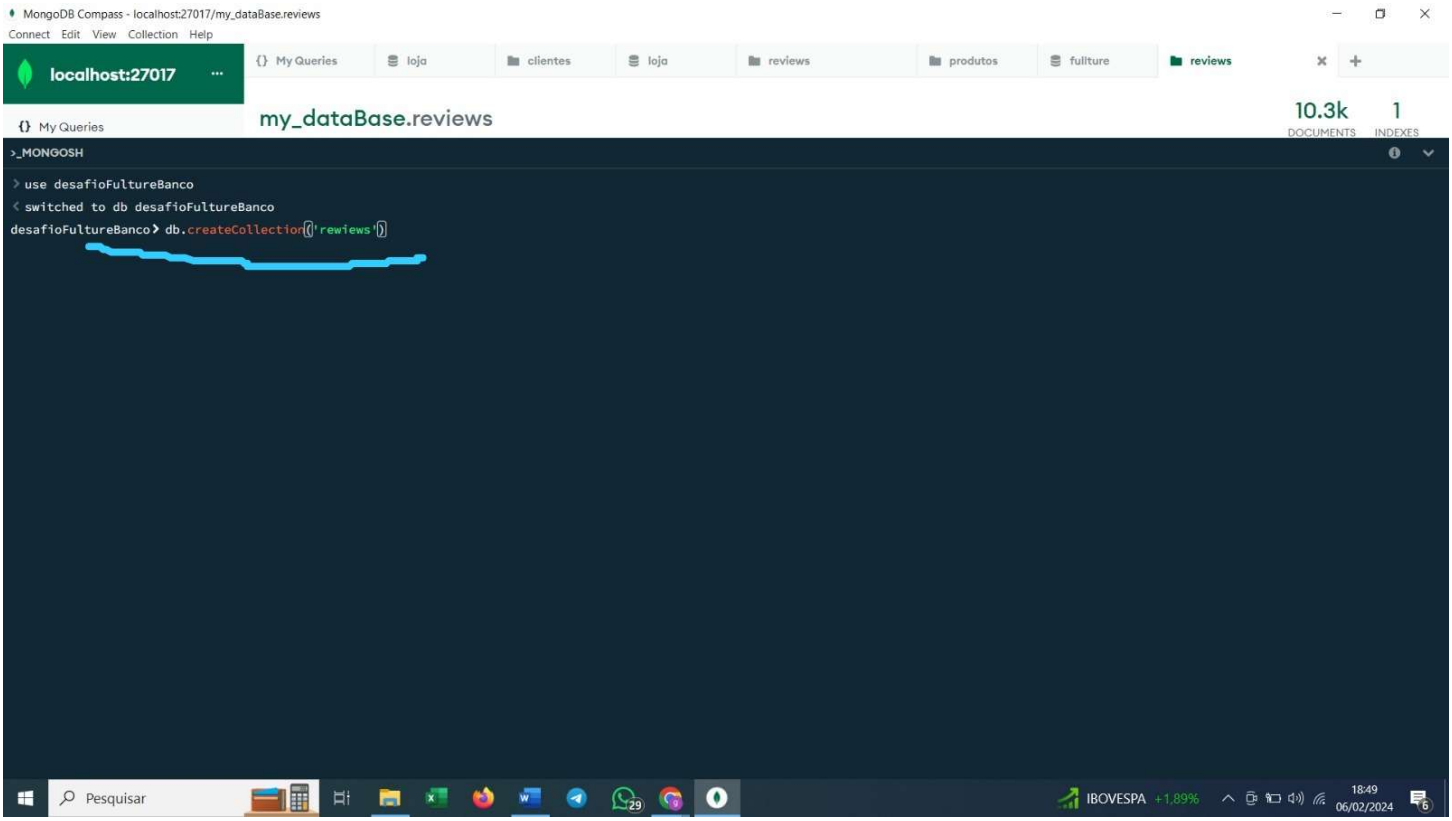
➢ Iniciando MongoDB. (segue img a baixo)



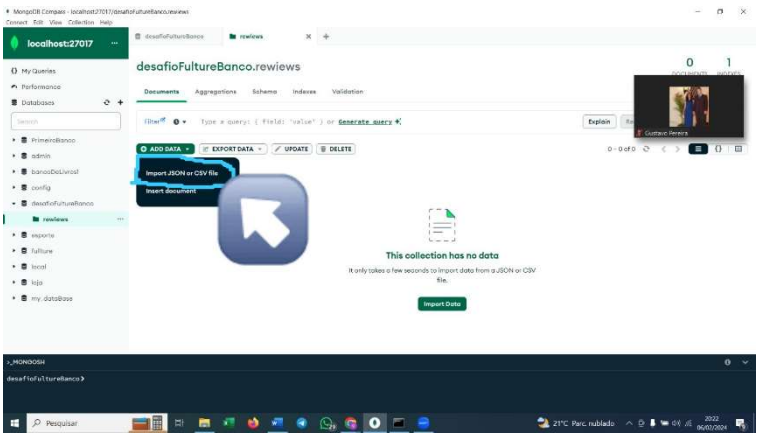
➢ Criando dataBase de forma implícita usando o comando use. (segue img a baixo)

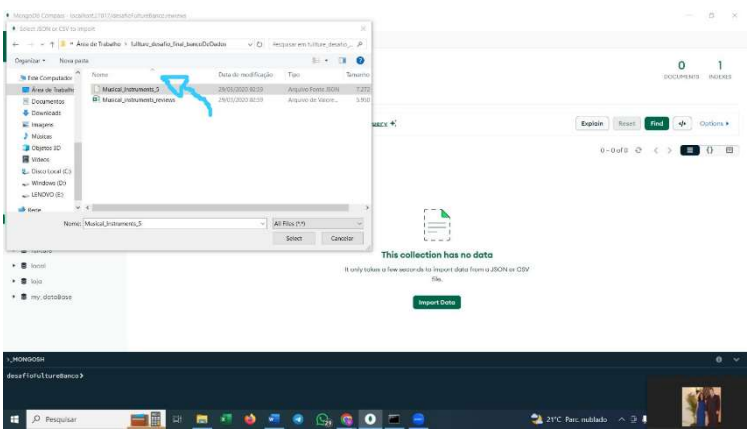


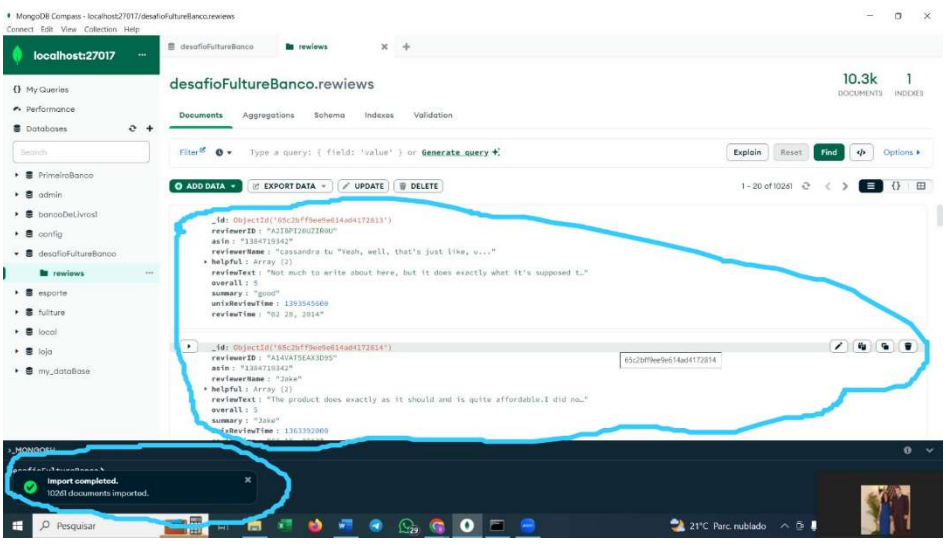
➢ Criando collection de forma explicita com o comando createCollect. (segue img a baixo)



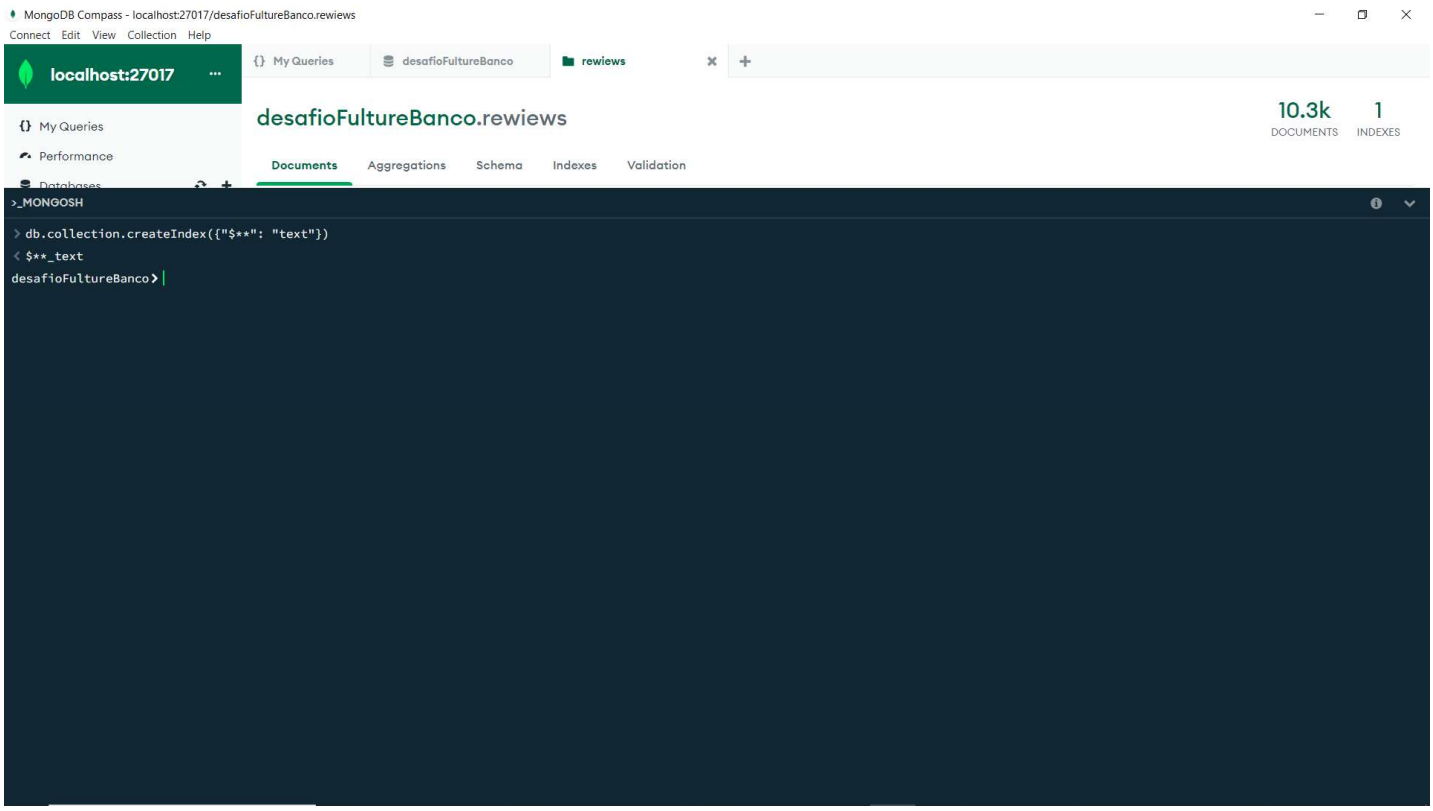
➢ Importando a dataSet de reviews da amazona no mongo . (segue img a baixo)



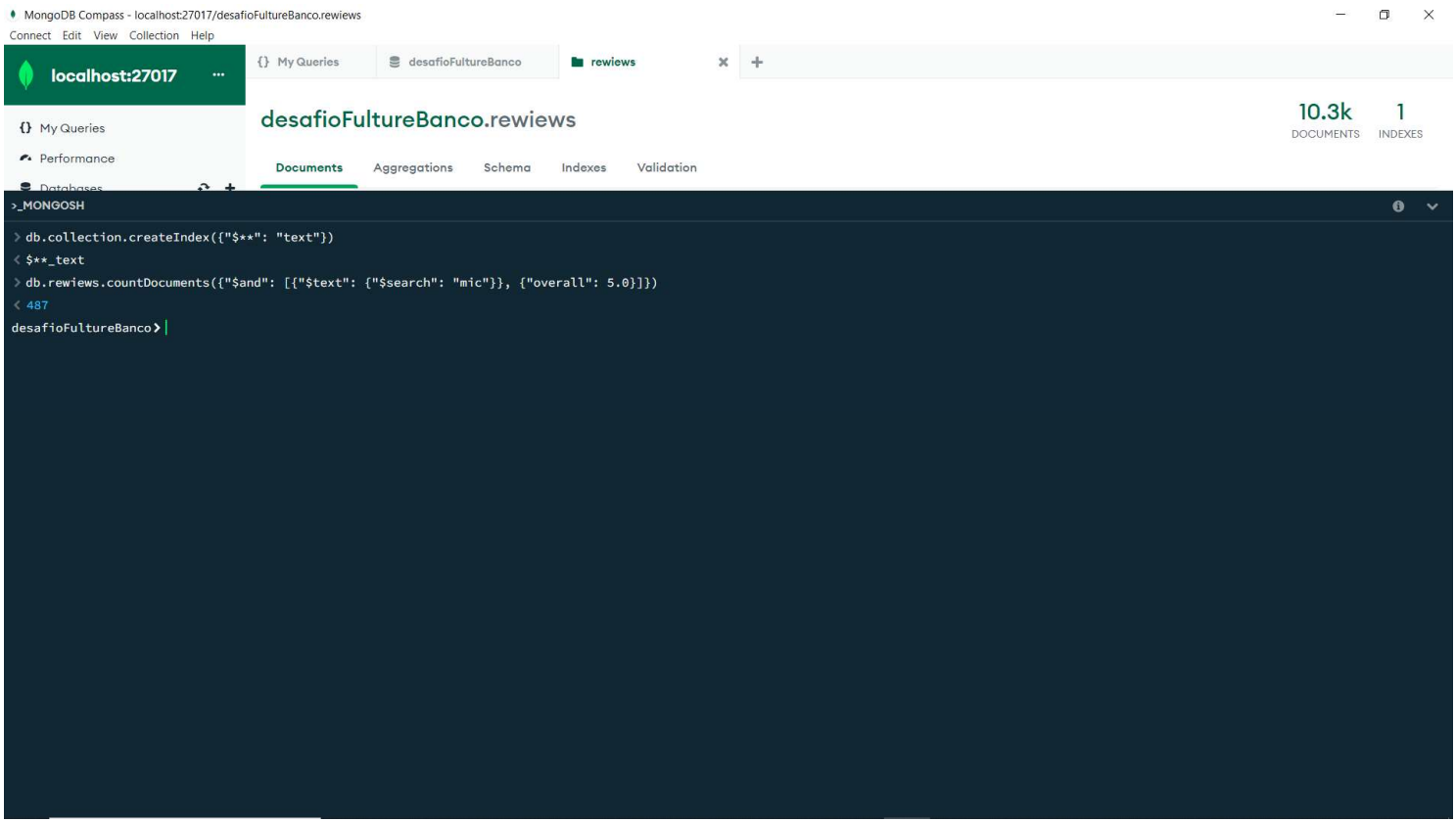




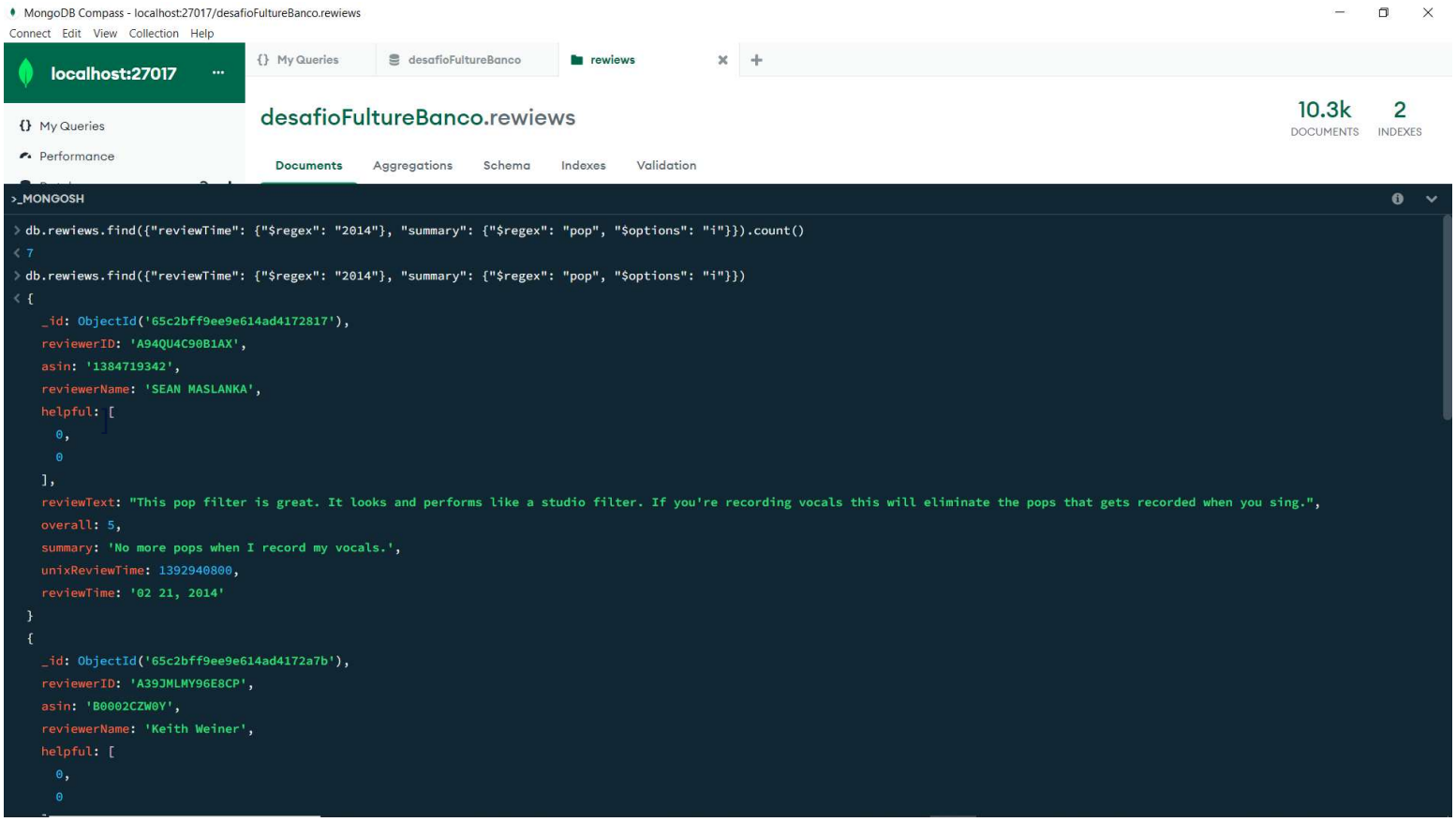
- criando um index de texto para melhorar a performace. (segue img a baixo)



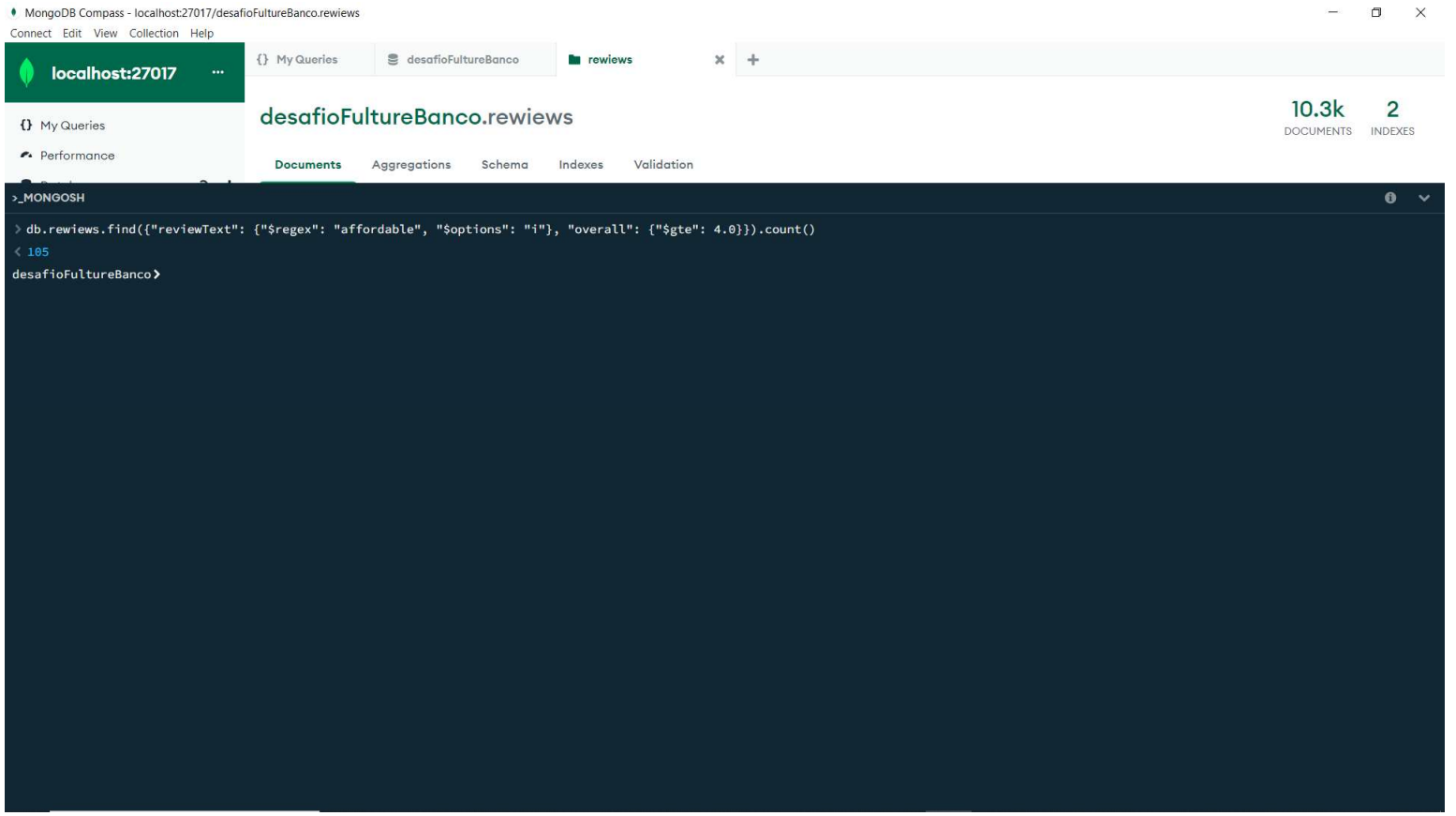
- Pesquisando no index todos os comentários que tenham "mic" e têm um overall de 5.0, usando o countDocuments ao invés do .count (segue img a baixo)



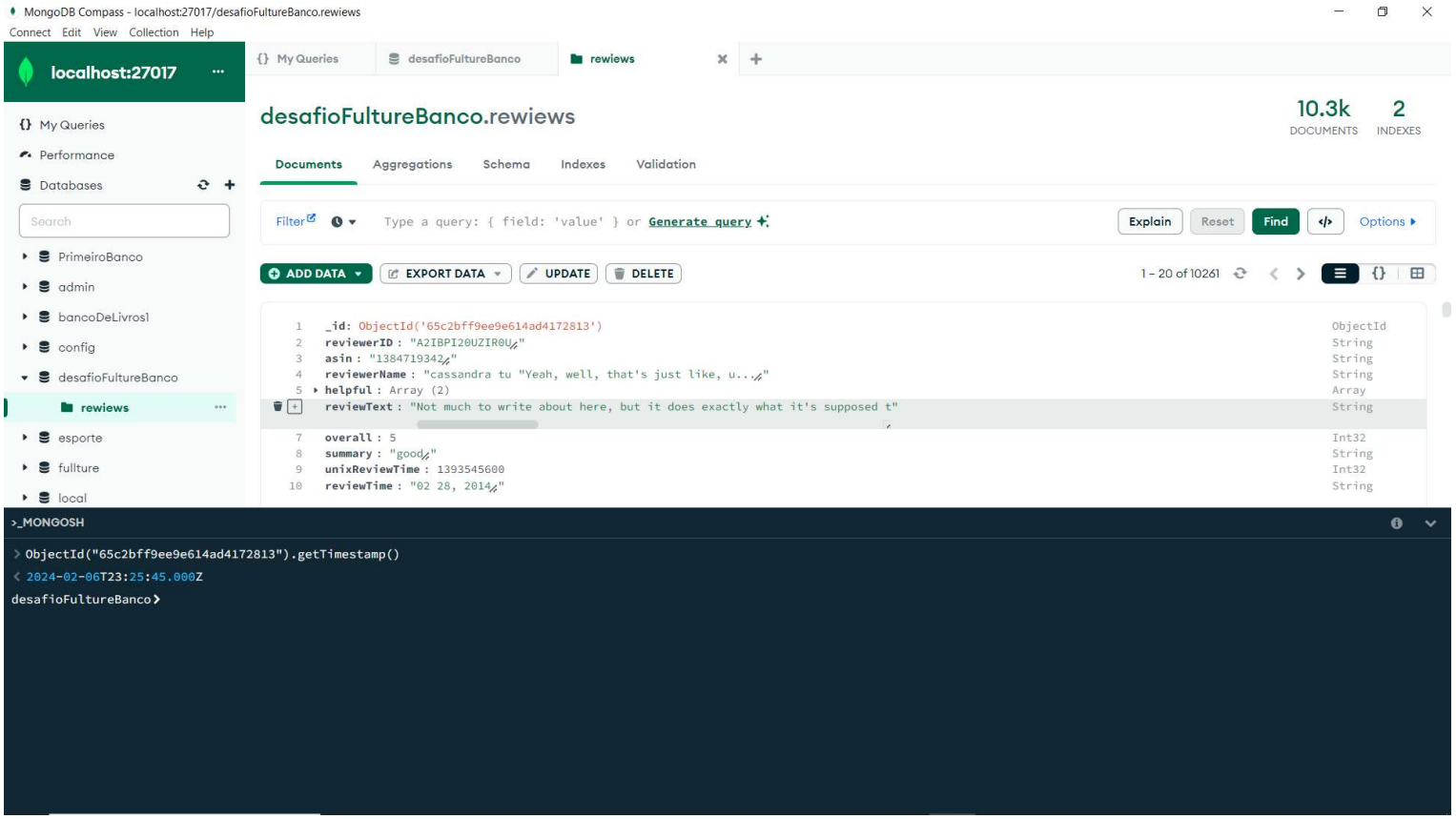
- Pesquisando reviews feitas em 2014 e que com a palavra pop no summary. (segue img a baixo)



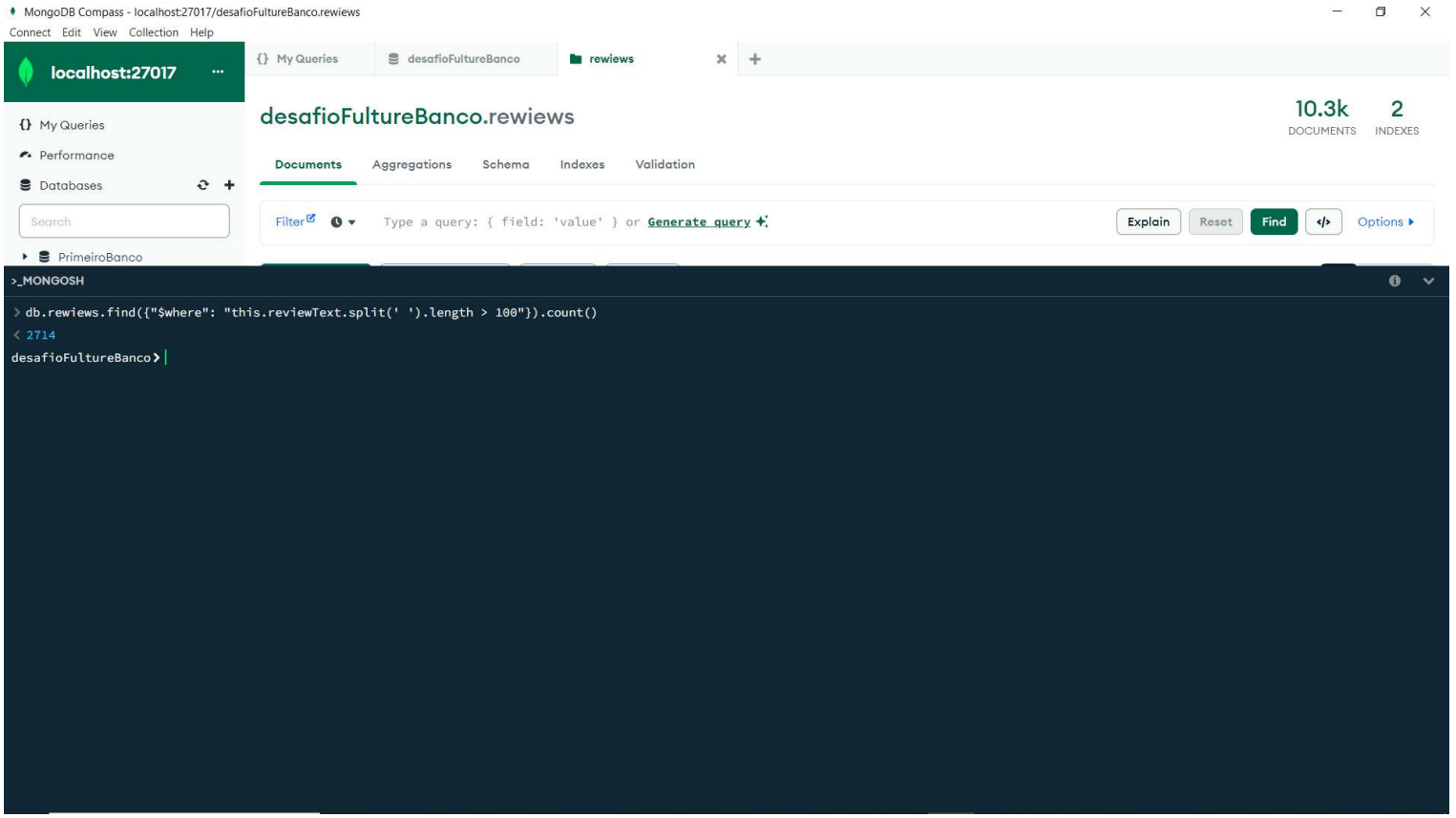
- Pesquisando reviews com a palavra affordable e com overall maior ou igual a 4.0. (segue img a baixo)



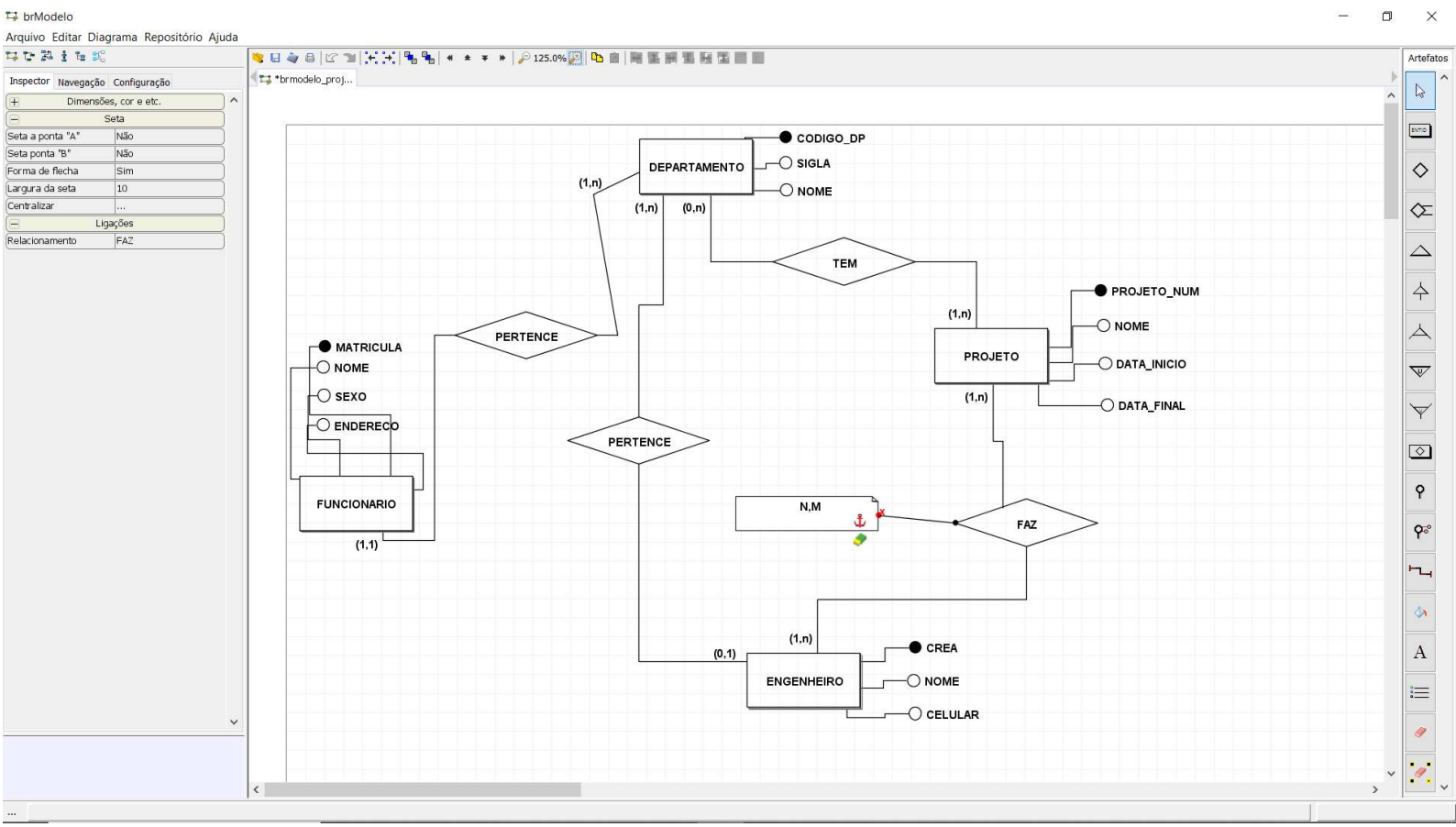
- Usando getTimestamp no primeiro objectId, para retornar data e hora da criação. (segue img a baixo)



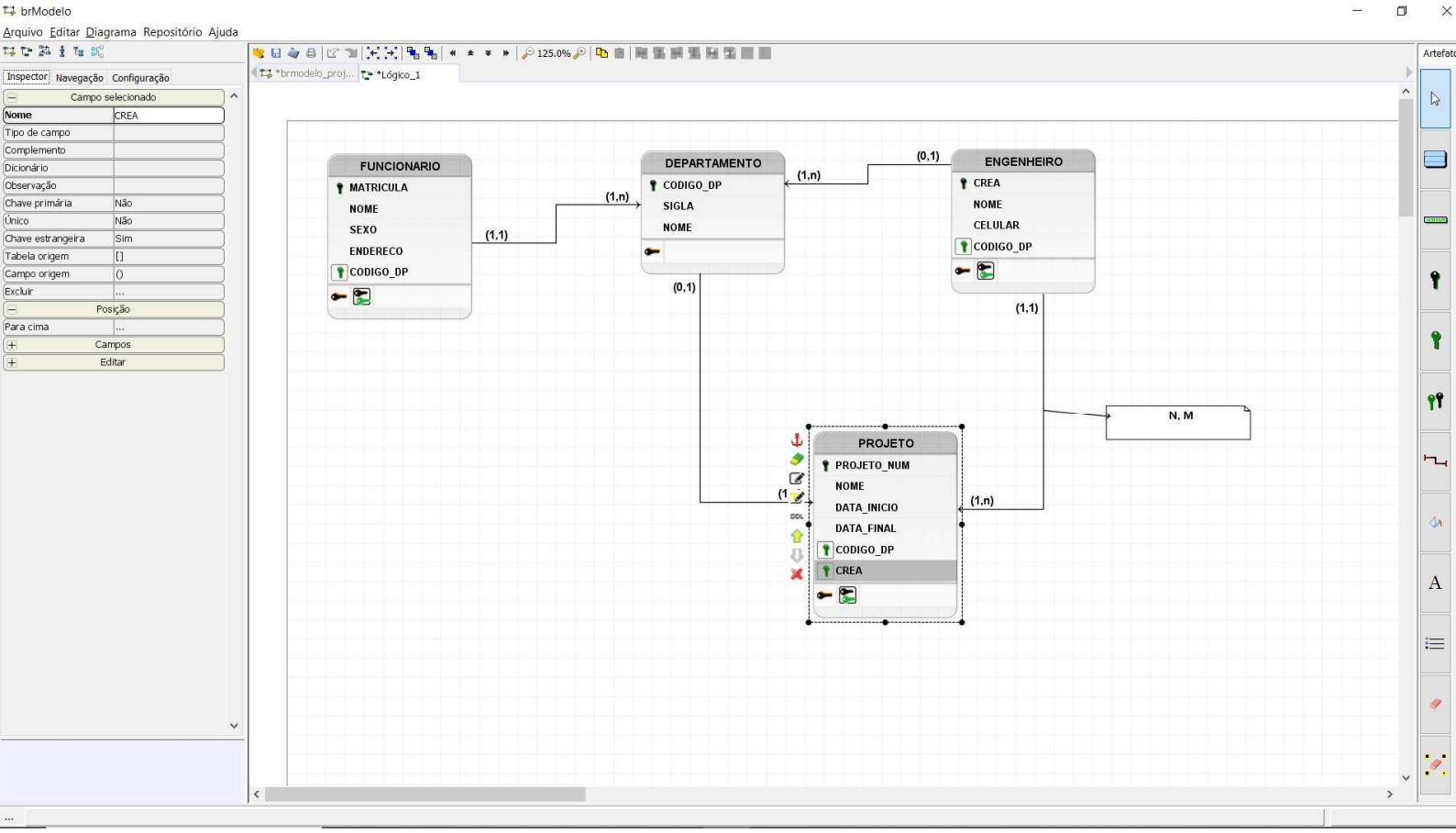
- Pesquisando reviewText com mais de 100 palavras. (segue img a baixo)



MODELO CONCEITUAL COM BRModelo.

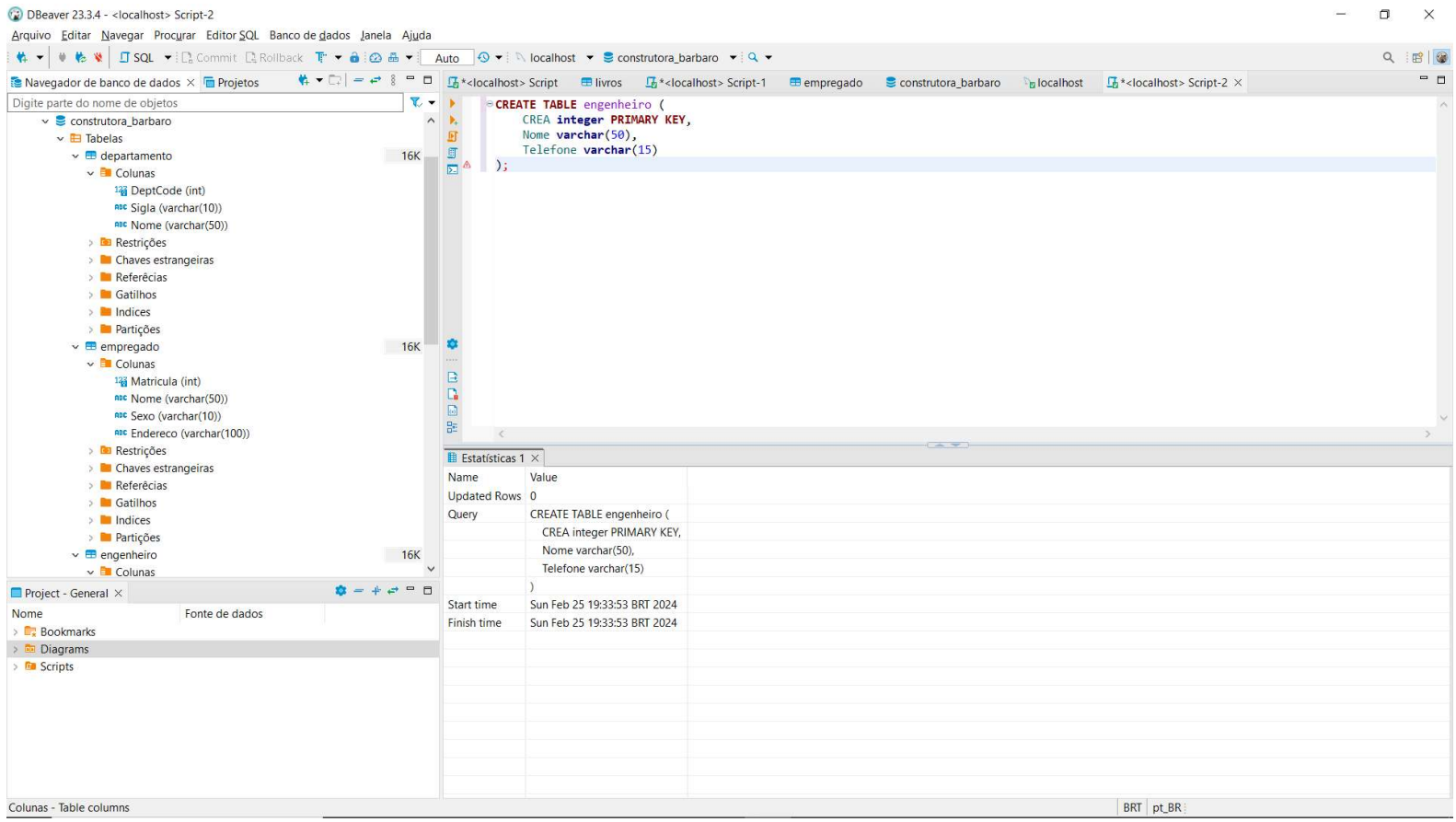


MODELO LOGICO COM BRModelo.



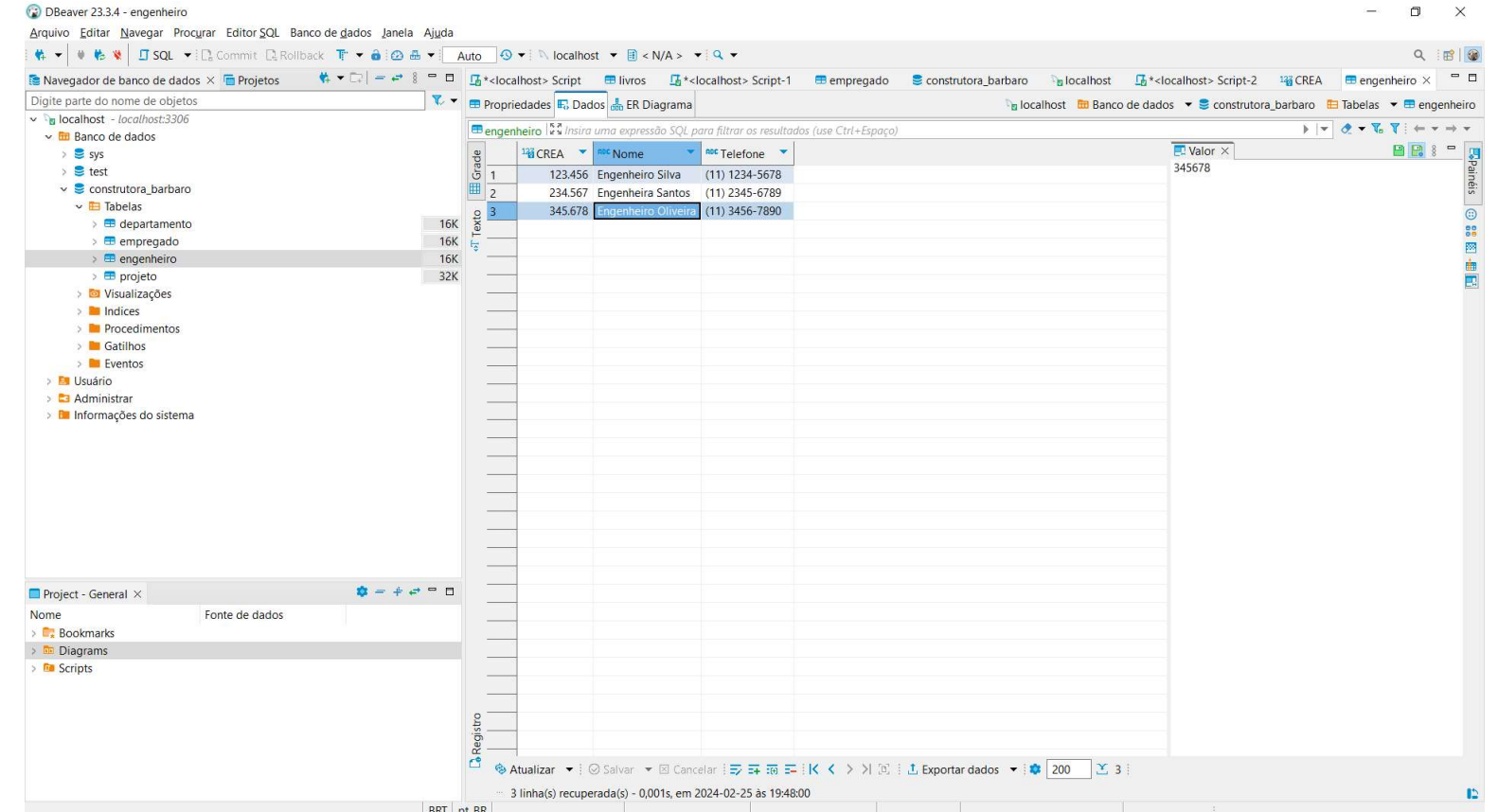
MODELO LINGUAGEM SQL COM DBeaver.

Criei um banco de dados construtora_barbaro e 4 tabelas. (Segue img a baixo)



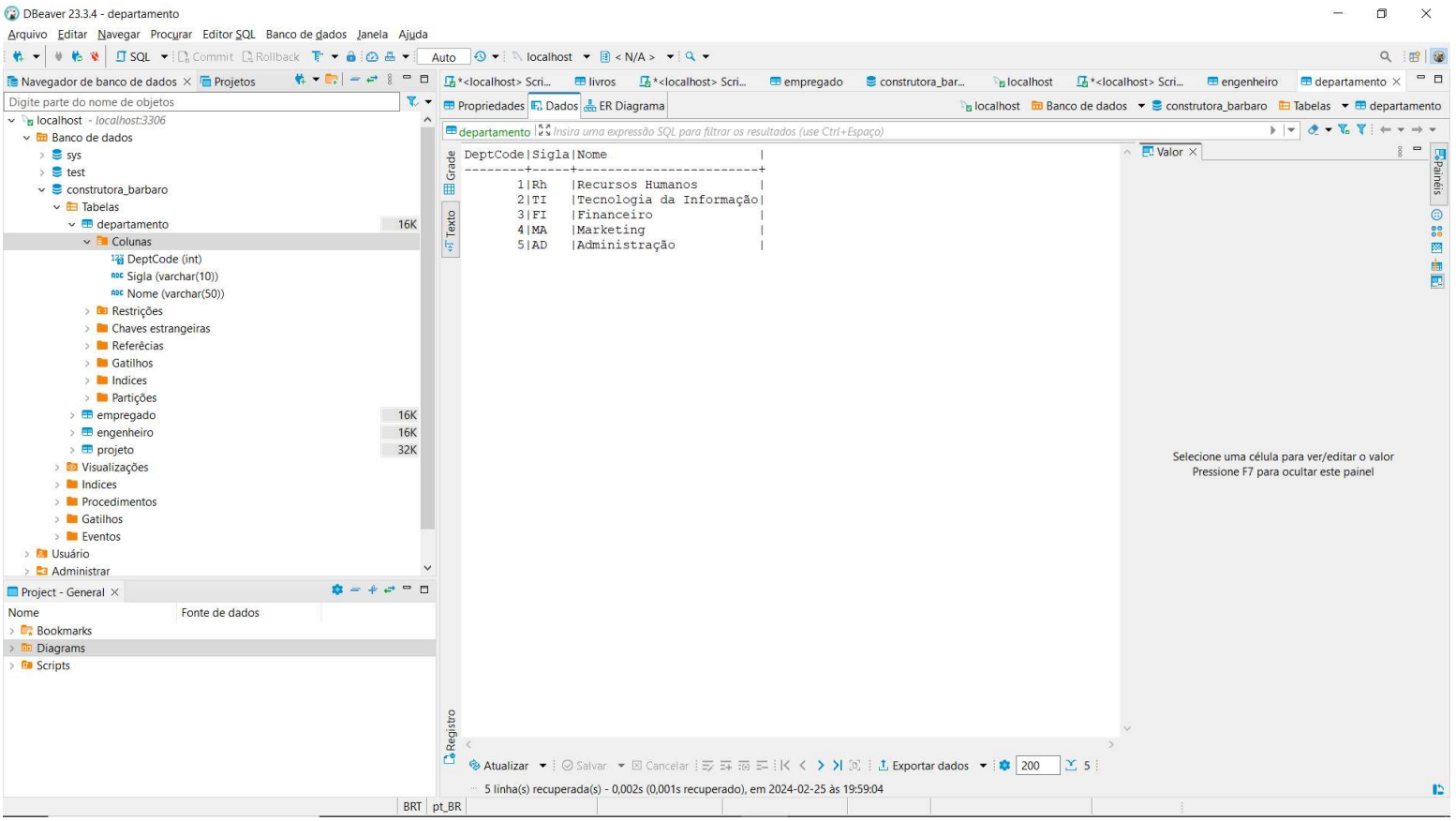
Inseri os dados na tabela engenheiro. (segue img a baixo)

```
INSERT INTO engenheiro (CREA, Nome, Telefone)
VALUES ('123456', 'Engenheiro Silva', '(11) 1234-5678');
```



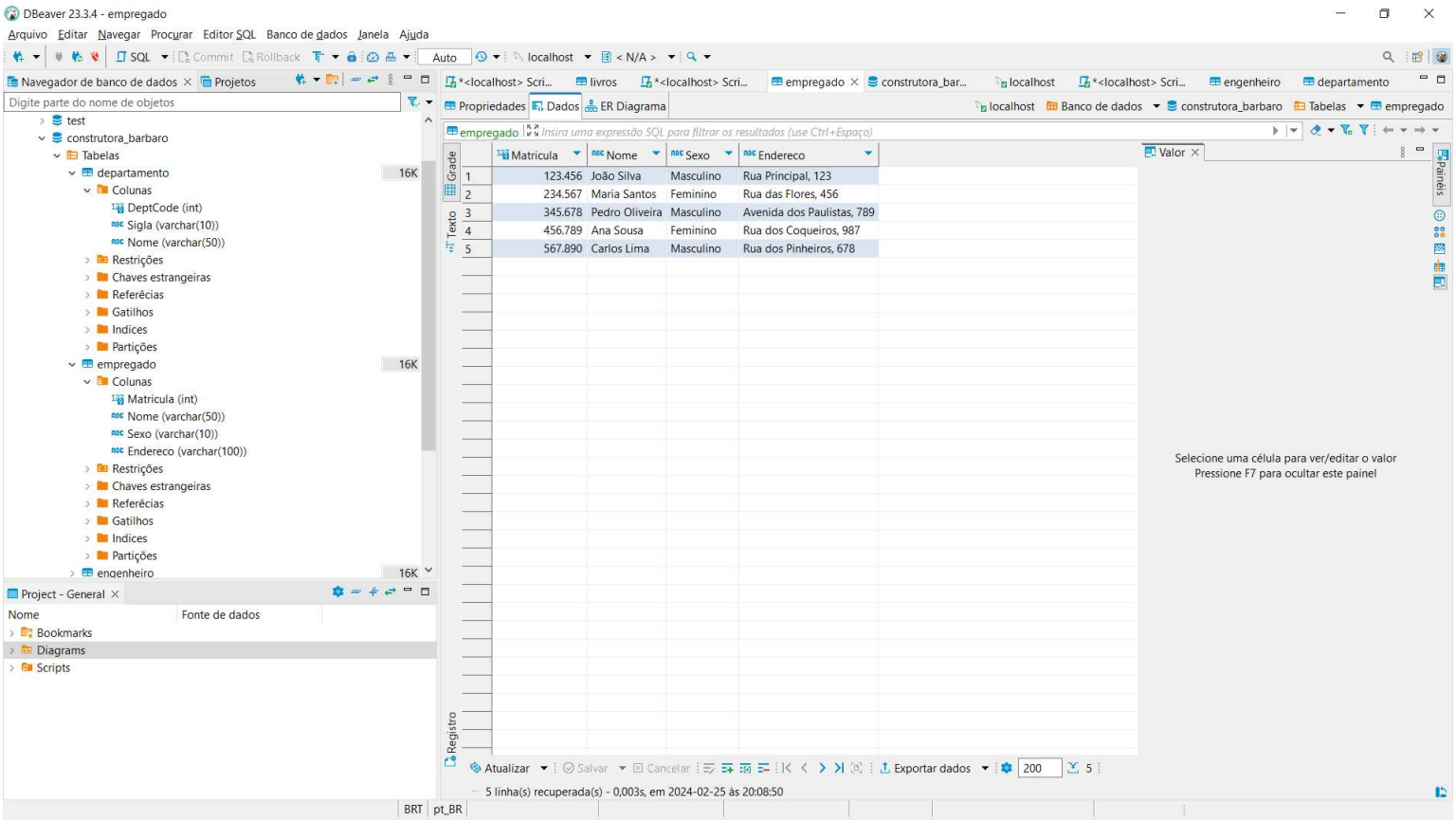
Inseri os dados na tabela departamento. (segue img a baixo)

```
INSERT INTO departamento (DeptCode, Sigla, Nome)
VALUES (5, 'AD', 'Administração');
```



Inseri os dados na tabela funcionario. (segue img a baixo)

```
INSERT INTO Empregado (Matricula, Nome, Sexo,  Endereco)
VALUES ('567890','Carlos Lima', 'Masculino',  'Rua dos Pinheiros, 678');
```



Inseri os dados na tabela projetos. (segue img a baixo)

```
INSERT INTO projeto (ProjCode, Nome, DataInicio, DataTermino)
VALUES (5, 'Projeto de Melhoria de Processos', '2023-09-01', '2023-12-31');
```