**2do Notes - MotoNotes**

Em um mundo cada vez mais agitado e repleto de informações, João, um desenvolvedor apaixonado por criar soluções impactantes, percebeu que, apesar de existirem diversas aplicações de anotações, muitas delas eram genéricas e não atendiam às necessidades específicas dos usuários modernos. Ele observou amigos, familiares e colegas de trabalho saltando de um aplicativo para outro, tentando juntar fragmentos de suas vidas digitais. Desde a estudante universitária que tentava organizar suas anotações de aula e resumos de livros, até o chef de cozinha que procurava uma maneira eficiente de catalogar suas receitas, todos enfrentavam o mesmo desafio: a falta de uma plataforma unificada que pudesse acomodar uma variedade tão rica de notas.

E assim nasceu a ideia da "MotoNotes". Uma aplicação que não só permitiria aos usuários criar notas tradicionais, mas que também ofereceria modelos personalizados para os 20 tipos mais comuns de anotações, desde planejamentos de viagem até meditações profundas. João imaginou uma interface intuitiva onde os usuários poderiam facilmente alternar entre diferentes tipos de notas, com etiquetas coloridas para categorização e um sistema robusto de histórico para rastrear cada alteração. Além disso, a inclusão de modelos predefinidos auxiliaria os usuários a iniciar suas anotações, proporcionando inspiração e estrutura.

Com a visão da MotoNotes claramente delineada em sua mente, João se lançou ao desafio de desenvolvimento. Ele acreditava firmemente que, ao atender às necessidades específicas dos usuários e ao mesmo tempo proporcionar flexibilidade, sua aplicação transformaria a maneira como as pessoas interagem e se organizam no mundo digital. E, à medida que a MotoNotes ganhava vida, ele estava prestes a descobrir o quão certa era essa intuição.

1. Listas de Compras: Anotações sobre o que comprar no supermercado ou outras lojas.

2. Lembretes: Notas que servem como lembretes de tarefas a serem realizadas.

3. Anotações de Reunião: Detalhes e pontos discutidos em reuniões.

4. Diário: Reflexões e experiências diárias.

5. Metas: Lista de objetivos a serem alcançados em um dia, semana, mês ou ano.

6. Planejamento de Viagem: Itinerário, detalhes do voo, lugares a visitar.

7. Ideias Aleatórias: Pensamentos ou ideias que surgem e que alguém pode querer lembrar.

8. Anotações de Aula: Conteúdo abordado em aulas, palestras ou webinars.

9. Receitas: Ingredientes e passo a passo de pratos culinários.

10. Orçamento e Finanças: Rastreamento de gastos, planejamento financeiro e contas a pagar.

11. Exercícios e Treinos: Planos de treino ou exercícios a serem feitos.

12. Livros, Filmes e Músicas: Listas de livros a ler, filmes a assistir ou músicas a ouvir.

13. Meditações e Reflexões: Pensamentos profundos ou insights.

14. Projetos: Planejamento e etapas para projetos pessoais ou profissionais.

15. Hábitos: Acompanhamento de hábitos diários ou metas de comportamento.

16. Resumos: Sumários de livros, artigos ou outros conteúdos.

17. Eventos e Programações: Detalhes de eventos futuros ou planos de programação.

18. Anotações de Pesquisa: Informações coletadas durante a pesquisa sobre um tópico específico.

19. Citações e Frases: Citações inspiradoras ou frases memoráveis.

20. Desenhos e Doodles: Alguns preferem expressar suas ideias e pensamentos através de esboços rápidos e rabiscos.

**Os Desafios de João com NoSQL**

Em meio ao fervor do desenvolvimento de sua inovadora aplicação, "MotoNotes", João se deparou com um obstáculo inesperado. Apesar de sua vasta experiência em programação e em bancos de dados relacionais, ele percebeu que o mundo dos bancos de dados NoSQL era um território não mapeado para ele. As vantagens dessas tecnologias, como flexibilidade de schema, escalabilidade e rapidez, pareciam ser ideais para sua aplicação, que visava acomodar uma variedade de tipos de notas e se adaptar constantemente às necessidades dos usuários. No entanto, ele rapidamente percebeu que modelar dados em NoSQL era uma arte distinta, exigindo uma abordagem diferente daquela dos sistemas relacionais tradicionais.

Cada vez que João tentava adaptar o design de seu banco de dados para acomodar uma nova funcionalidade, ele se sentia como um navegante perdido em um oceano vasto e desconhecido. Era evidente que a flexibilidade dos bancos de dados NoSQL vinha com suas próprias complexidades. A multiplicidade de opções, desde a escolha entre tipos diferentes como document-based, column-based ou graph-based, até as nuances da modelagem de dados para otimizar consultas e garantir a integridade dos dados, era esmagadora.

Reconhecendo a importância de construir uma base de dados robusta e eficiente para sua aplicação, João decidiu buscar ajuda. Ele sabia que, com a orientação correta e uma compreensão mais profunda dos bancos de dados NoSQL, ele poderia não apenas superar seus desafios atuais, mas também garantir que a "MotoNotes" estivesse posicionada para o sucesso no longo prazo. E assim, com humildade e determinação, ele deu o próximo passo em sua jornada, em busca de um mentor ou especialista que pudesse guiá-lo pelas intricadas paisagens do NoSQL.

Ajude João a criar uma modelagem de banco de dados NoSQL chave-valor adequada para a aplicação "MotoNotes", levando em consideração os seguintes requisitos:

A aplicação deve suportar múltiplos usuários.

Cada usuário pode ter várias notas, e cada nota deve ter uma categoria ou tipo (por exemplo, lista de compras, lembretes, anotações de aula, etc.).

As notas devem ter a capacidade de conter tags ou marcadores.

É fundamental manter um histórico de alterações para cada nota.

A aplicação deve suportar a busca por notas baseada em tags, tipos ou conteúdo da nota.

Considerando a natureza chave-valor do banco de dados, pense em como as chaves e os valores serão estruturados para obter a máxima eficiência.

Instruções:

Comece identificando as principais entidades e seus relacionamentos.

Determine uma estratégia de nomenclatura de chaves que facilite consultas rápidas e eficientes.

Descreva como os valores serão armazenados e estruturados para cada tipo de chave.

Pense em possíveis otimizações, como a utilização de chaves compostas ou o uso de índices secundários, se a plataforma NoSQL escolhida suportar.

Elabore um breve documento descrevendo sua proposta de modelagem, justificando suas escolhas e fornecendo exemplos de como as operações típicas (como inserção, atualização, busca e exclusão) seriam realizadas no banco de dados proposto.

Lembre-se de que o principal benefício dos bancos de dados NoSQL chave-valor é sua simplicidade e velocidade, portanto, sua modelagem deve refletir essas características. Boa sorte!

* **Problema a ser solucionado**

João tem uma vasta experiencia com banco de dados relacionais, porém para o desenvolvimento de sua inovadora aplicação “MotoNotes”, ele percebeu que seria melhor a utilização do NoSQL, entretanto não sabia como utilizá-lo, e para isso decidiu contratar a nossa equipe para ajudá-lo a migrar todos os dados do banco de dados relacional para o não relacional.

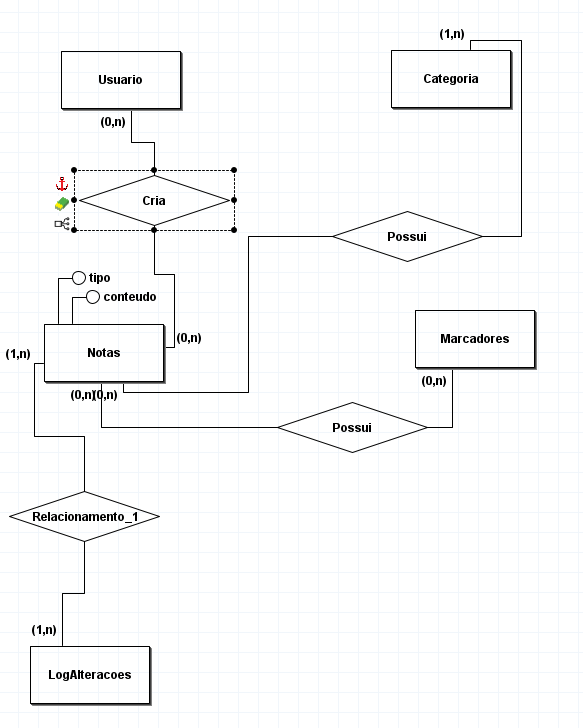
* **Método aplicado**

Para o desafio proposto, é necessária a adequação de um modelo de Banco de Dados NoSQL à armazenagem de vários tipos de notas autoadesivas digitais, de acordo com a necessidade de cada usuário, sendo obrigatória a alternação entre categorias de notas.

Nesse contexto, foi definida uma modelagem inicial entre entidade e relacionamentos, como também a lógica, a qual os dados seguirão para o correto funcionamento da aplicação.

* **Solução desenvolvida**

O relacionamento entre usuário e outras tabelas deverão ser por uma chave única, implicando também no link com as notas via índice único, vinculando usuário a nota.



**Foram aplicadas, de acordo com a imagem relacional acima, as seguintes soluções:**

* O banco possuirá capacidade de armazenamento de vários usuários;
* Cada usuário poderá criar várias notas autoadesivas para organização de suas informações de diferentes maneiras;
* Cada usuário, possui como atributos, nome e e-mail;
* Cada nota autoadesiva contém pelo menos uma categoria, podendo ela, ter várias;
* Cada nota autoadesiva contém pelo menos uma tag, podendo ela, ter várias;
* Ao se criar e modificar uma nota, ela gerará um arquivo log salvo no banco, contendo o histórico de cada alteração de suas notas.
* Cada nota, possui como atributos, tipo e conteúdo;
* O arquivo Log, gerará como resultado, a data de alteração da nota, e o conteúdo anterior, antes de sua modificação;

**Utilização de modelagem NoSQL, com base em documentos chave-valor:**

Para o exercício proposto, optamos por desenvolver um Banco de Dados orientado a documentos, para que sejam organizados em um modelo chave-valor, de fácil acesso e manipulação, como também, fácil para processamento do servidor.

{

"usuarios": [

{

"id": 1,

"nome": "Nome do Usuário",

"email": "email@example.com"

}

]

}

{

"notas": [

{

"id": 1,

"tipo": "Tipo da Nota",

"conteudo": "Conteúdo da Nota",

"id\_usuario": 1

}

]

}

{

"categorias": [

{

"id": 1,

"nome": "Nome da Categoria",

"id\_nota": 1

}

]

}

{

"marcadores": [

{

"id": 1,

"nome": "Nome do Marcador",

"id\_nota": 1

}

]

}

{

"log\_alteracoes": [

{

"id": 1,

"descricao": "Descrição da Alteração",

"data": "2023-09-16",

"id\_nota": 1

}

]

}

Criamos essa modelagem para armazenar informações sobre usuários, notas, categorias, marcadores e registros de alterações

* Chave usuários possui informações sobre usuários, identificador único e dados de identificações.
* Chave notas possui informações sobre as notas e os objetos dele possui identificador único, o tipo de nota e conteúdo.
* Chave categorias contêm informações das categorias e os objetos id único e nome.
* Chave marcador contém informações sobre a representação do marcador id único e nome.
* Chave log possui as informações sobre as alterações, cada objeto representa as alterações feitas, possuem os campos id único, descrição e data.
* Chave notas lincadas com categorias, relaciona notas as categorias via id nota e id da categoria.
* Chave notas lincadas aos marcadores, relaciona notas aos marcadores via id nota e id do marcador.
* Chave notas lincadas aos logs, os objetos possuem os campos de ligação id da nota e log id alterações para referenciar a indicação de modificação a seu autor.

Devemos priorizar consultas, indexação dos dados do banco e sua segurança.

**Operações CRUD com o NoSQL:**

* **Insert**: A cada novo usuário cadastrado, um novo valor na chave “Usuários” será acrescentado. Para cada nota criada por usuário, serão preenchidas as chaves - “Notas(Possuem a chave id\_usuario)”, “Marcadores(Possuem a chave id\_nota)” e “Categorias(Possuem a chave id\_nota)” - Cada nota criada e alterada, terá um log com seu id e informações salvos.
* **Update**: Quando alterados as informações do usuário, somente o valor de sua chave respectiva será alterado. Quando alteradas as notas de cada usuário, os marcadores e categorias poderão ser alterados devido ao id\_notas linkados em suas chaves;
* **Delete**: Quando deletados registros de notas de um determinado usuário, as informações salvas da nota, suas categorias e tags, serão apagadas, por terem a chave “id\_nota” em suas estruturas. Quando um usuário é deletado, suas notas serão apagadas, pois em sua estrutura está presente a chave “id\_usuario”;
* **Select**: Quando selecionados dados de um usuário, todas as notas criadas por ele, serão exibidas, por conter a chave “id\_usuario” e por consequência as características de cada nota serão mostradas, por possuir sua chave “id\_nota” nas chaves “Marcadores” e “Categorias”.