**Banco de Dados Não Relacional**

Nós escolhemos o MongoDB pois é uma ótima opção para muitos casos de uso, especialmente em cenários onde a escalabilidade, a flexibilidade e o alto nível de consumo de dados são prioridades. No entanto, é essencial entender suas nuances e considerar cuidadosamente os requisitos específicos do projeto ao escolher um banco de dados.

**Vantagens do MongoDB:**

1. **Escalabilidade Horizontal:** O MongoDB é altamente escalável horizontalmente, permitindo lidar facilmente com grandes volumes de dados distribuindo-os em vários servidores.
2. **Esquema Flexível:** Sua estrutura de dados é flexível e dinâmica, permitindo armazenar diferentes tipos de dados sem a necessidade de um esquema rígido.
3. **Desempenho e Velocidade:** É conhecido por seu desempenho rápido, especialmente para operações de leitura, devido à sua capacidade de indexação eficiente.
4. **Replicação e Tolerância a Falhas:** Oferece recursos robustos de replicação para garantir alta disponibilidade e tolerância a falhas.
5. **Consultas Ricas:** Suporta consultas complexas e permite o uso de operadores de consulta poderosos.

**Desvantagens e Limitações:**

1. **Consistência eventual:** Em ambientes distribuídos, o MongoDB pode fornecer consistência eventual por padrão, o que significa que nem sempre garante a leitura mais recente dos dados.
2. **Consumo de Recursos:** Em comparação com alguns bancos de dados relacionais, o MongoDB pode consumir mais recursos de hardware, especialmente em cenários de gravação intensiva.
3. **Fragmentação:** À medida que os dados aumentam, podem ocorrer fragmentações de disco e lentidão nas consultas, se não forem tratadas adequadamente.
4. **Complexidade de Consulta:** Consultas muito complexas podem ser difíceis de otimizar e podem levar a problemas de desempenho.
5. **Curva de Aprendizado:** Para aqueles mais familiarizados com bancos de dados relacionais, a transição para um banco de dados NoSQL como o MongoDB pode exigir uma curva de aprendizado.

Por esse motivo escolhemos o mongo DB para usamos no nosso projeto.

**A modelagem é a seguinte:**

O sistema terá cadastro de pessoas(beneficiados), cadastro de doadores, controle de estoque (controle de vencimento de alimentos), distribuiçao de alimentos.

Ex:

Um doador poderá inserir seus dados e inserir a quantidade de alimentos, como também alteração dos dados inseridos caso ele insira errado.

As pessoas(beneficiadas) pelas doações poderão fazer os cadastros com os seus dados, e poderão fazer o pedido para a doação, como também alteração de dados caso ele precise mudar algum endereço ou outro dado.

O controle de estoque sera registrado conforme os doadores forem colocando seus alimentos doados, e quem ficara responsável por algum evento anormal sera o adm, podendo excluir cadastros de ambos, tanto doadores ou pessoas beneficiadas, tanto quanto a modificação dos seus dados.

Modelagem de exemplo (não está completa, haverá alteração).

const doador = {

  doador\_id: "UUID-Doador",

  nome: "Nome do Doador",

  contato: "Informações de Contato do Doador"

};

const doacao = {

  doacao\_id: "UUID-Doacao",

  doador\_id: "UUID-Doador",

  data\_doacao: "Data da Doação",

  alimentos: [

    {

      alimento\_id: "UUID-Alimento1",

      tipo: "Tipo de Alimento 1",

      quantidade: "Quantidade doada 1",

      validade: "Data de Validade 1"

    },

    {

      alimento\_id: "UUID-Alimento2",

      tipo: "Tipo de Alimento 2",

    quantidade: "Quantidade doada 2",

      validade: "Data de Validade 2"

    }

  ]

};

const doador = {

  doador\_id: "UUID-Doador",

  nome: "Nome do Doador",

  contato: "Informações de Contato do Doador"

};

const estoque = {

  estoque\_id: "UUID-Estoque",

  alimentos: [

    {

      alimento\_id: "UUID-Alimento3",

      tipo: "Tipo de Alimento 3",

      quantidade: "Quantidade em Estoque 1",

      validade: "Data de Validade 3"

    },

    {

      alimento\_id: "UUID-Alimento4",

      tipo: "Tipo de Alimento 4",

      quantidade: "Quantidade em Estoque 2",

      validade: "Data de Validade 4"

    }

  ]

};

const distribuicao = {

  distribuicao\_id: "UUID-Distribuicao",

  organizacao\_id: "UUID-Organizacao",

  data\_distribuicao: "Data da Distribuição",

  alimentos\_distribuidos: [

    {

      alimento\_id: "UUID-Alimento1",

      tipo: "Tipo de Alimento 1",

      quantidade: "Quantidade Distribuída 1"

    },

    {

      alimento\_id: "UUID-Alimento2",

      tipo: "Tipo de Alimento 2",

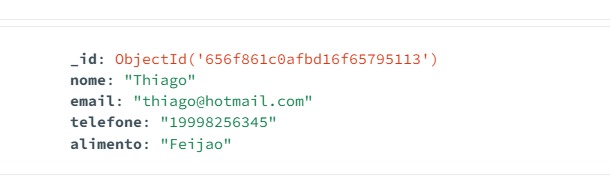
      quantidade: "Quantidade Distribuída 2"

    }

  ]};

Agora temos exemplo do uso de agregação no nosso banco de dados NoSQL.

Aqui é exemplo sem agregação:





Aqui é como ele retorna depois que usamos agregação:

