
UniSENAI
O FUTURO COMEÇA
POR VOCÊ!

UniSENAI

Modelo lógico

Modelo não relacional

Prof. Aline Rubenich, Ma.

Disciplina: Projeto e gerenciamento de banco de dados

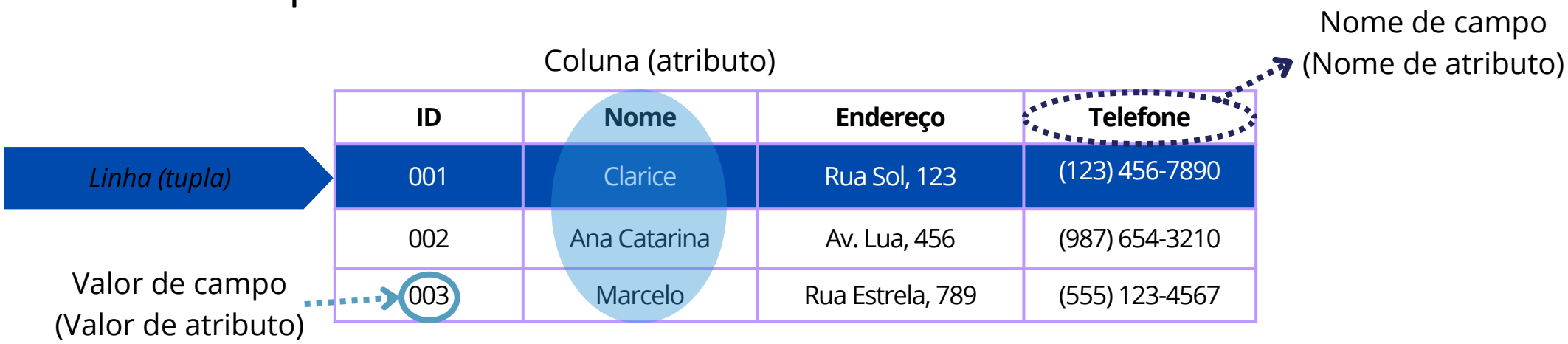
Curso: Tecnologia em Análise e Desenvolvimento de Sistemas

Revisão

Modelo relacional (MR): é um modelo lógico fundamentado em **registros**.

Mapeamento: uma **migração** de um modelo para o outro, ou seja, do MER para o MR.

Tabela: São os elementos da **modelagem** MER. Um conjunto de linhas, também conhecido como campos, e a interseção entre linhas e colunas é chamada de tupla.



ID	Nome	Endereço	Telefone
001	Clarice	Rua Sol, 123	(123) 456-7890
002	Ana Catarina	Av. Lua, 456	(987) 654-3210
003	Marcelo	Rua Estrela, 789	(555) 123-4567

Revisão

Chaves: São atributos ou conjuntos de atributos que identificam de forma única cada tupla em uma tabela de banco de dados. Existem classificações de chaves:

- Chave primária: Identifica de forma única cada tupla em uma tabela.
- Chave estrangeira: Estabelece uma relação entre duas tabelas, referenciando a chave primária de outra tabela.
- Chave alternativa: Poderia ser escolhido como chave primária se a chave primária atual não existisse.

Atributo multivalorado

Um tipo de atributo que pode ter vários valores ligados a uma única entidade em um banco de dados. É preciso criar uma nova tabela para os telefones, pois na tabela Alunos há espaço somente para um valor de telefone caso houvesse um atributo simples.

Alunos

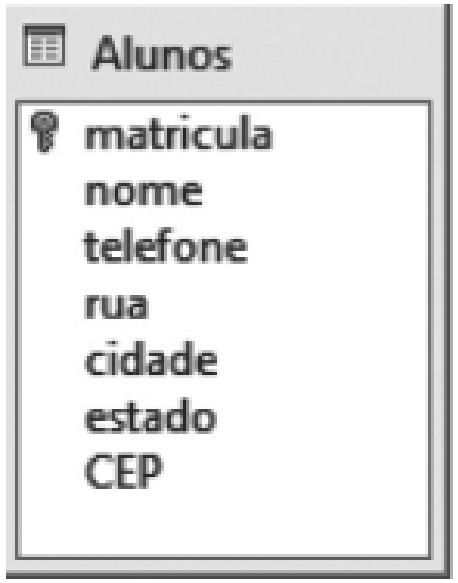
matrícula	nome	endereço
1023457	João Alencar Souza	R. Das Alamedas 67 - São Paulo
1098234	Maria Luiza Pontes	Pça. Da Bandeira 1.234 - São Paulo
1022764	Sara Bitencourt	R. 13 de Maio 345 - São Paulo
1024987	André Luis Paes	Av. Ambrósia 1.765 - São Paulo
1032322	Paulo da Silva	R. Paiva Oliveira 211 - São Paulo

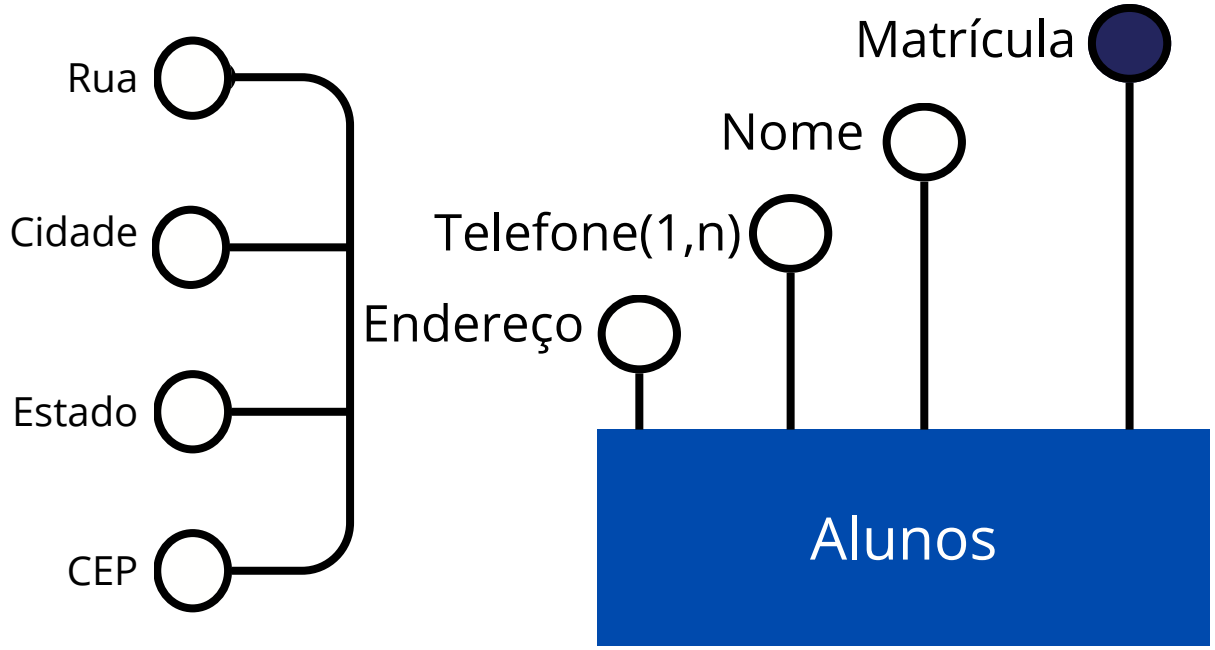
Telefone-Alunos

matrícula	telefone
1022764	(11)7312-9887
1022764	(11)8767-9121
1024987	(11)6511-9086
1023457	(11)7575-4322
1023457	(11)4311-2121
1023457	(11)6392-5972
1023457	(11)4821-3545

Atributo composto

Que nos permite indicar um atributo que pode ser dividido em outros, como no problema do endereço em que devemos indicar rua, cidade, estado e CEP.

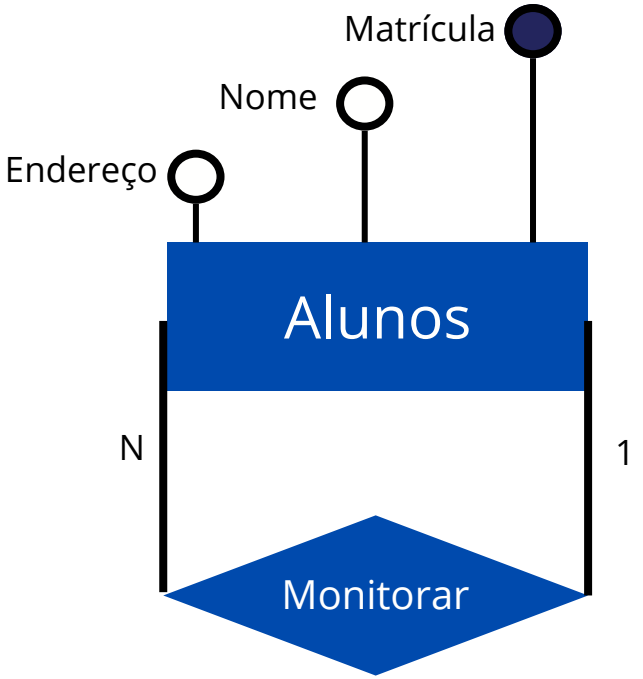




Relacionamento recursivo

Se o modelo apresentar um relacionamento recursivo ou autorrelacionamento, esse deve ser mapeado seguindo as mesmas regras, ou seja, de acordo com a cardinalidade.

Um aluno pode ser monitor da sua turma

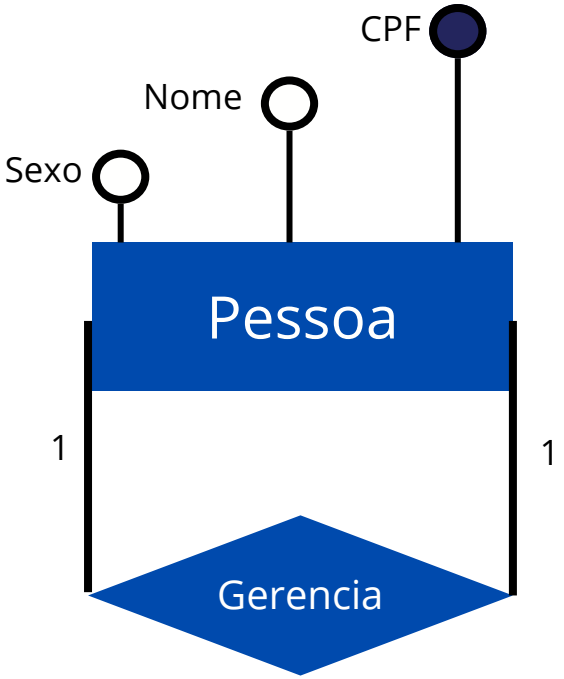


Matrícula	Nome	Endereço	Matrícula- monitor
896.574.896-55	Alice	R. da Paz, 50	745.987.152-96
745.987.152-96	Bryan	Av. da Flor, 60	896.574.896-55
785.695.745-85	João	R. Pereira,70	896.574.896-55
123.456.789-01	Francisco	Av. dos Ypês, 30	

Relacionamento recursivo

Outro exemplo

Uma pessoa gerencia pessoa

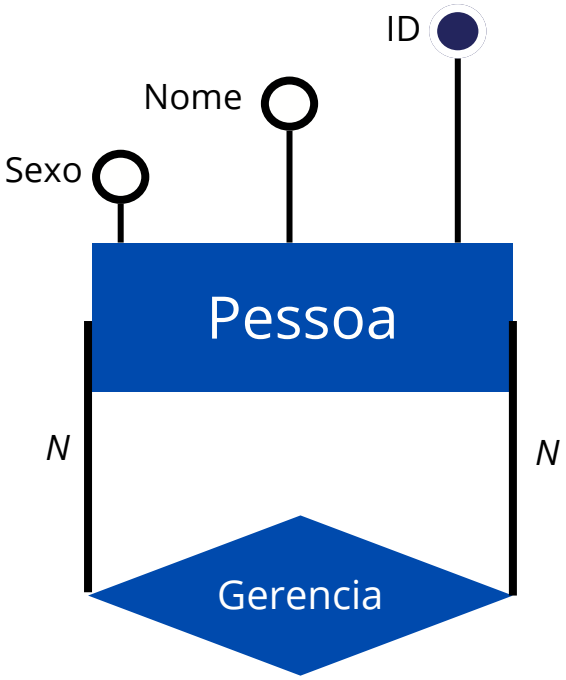


CPF	Nome	Sexo	P_CPF_Ger
123.456.789-01	Amanda	F	987.654.321-09
987.654.321-09	Carlos	M	555.888.777-33
555.888.777-33	Beatriz	F	

Relacionamento recursivo

Outro exemplo

Uma pessoa gerencia pessoa

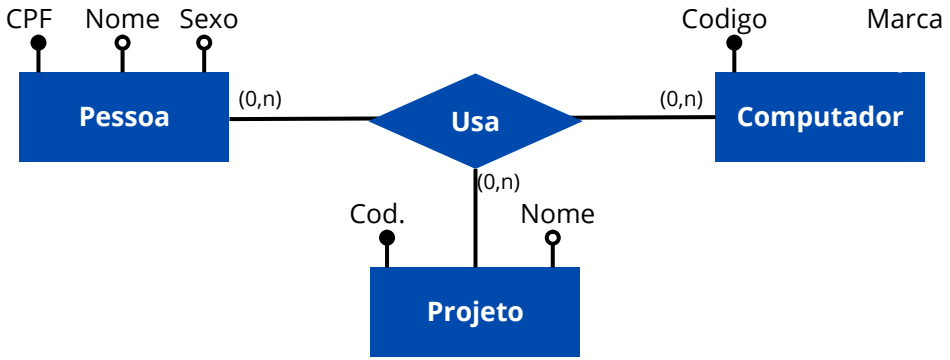


ID	Nome	Sexo
111	Amanda	F
222	Carlos	M
333	Beatriz	F
444	Jose	M

ID_gerente	ID_subordinado
111	222
111	333
222	333
333	444

Relacionamento ternário

Outro exemplo



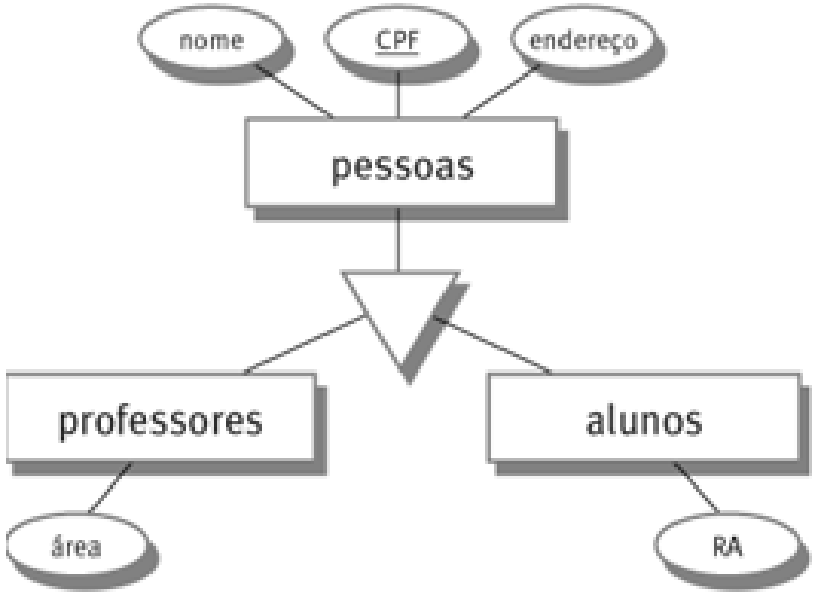
PESSOA		
CPF	**Nome**	**Sexo**
159.753.852-46	Amanda	F
789.456.123-96	Carlos	M

COMPUTADOR	
codigo	**Marca**
111	Dell
222	Positivo

USA		
P_CPF	**C_codigo**	**P_ID**
159.753.852-46	111	P01
159.753.852-46	222	P02
789.456.123-96	222	P01

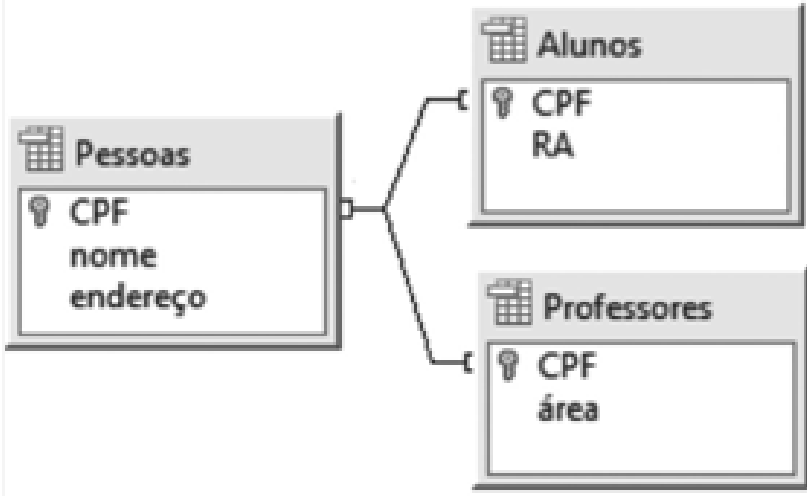
PROJETO	
ID	**Nome**
P01	Projeto X
P02	Projeto Y

Especialização/generalização



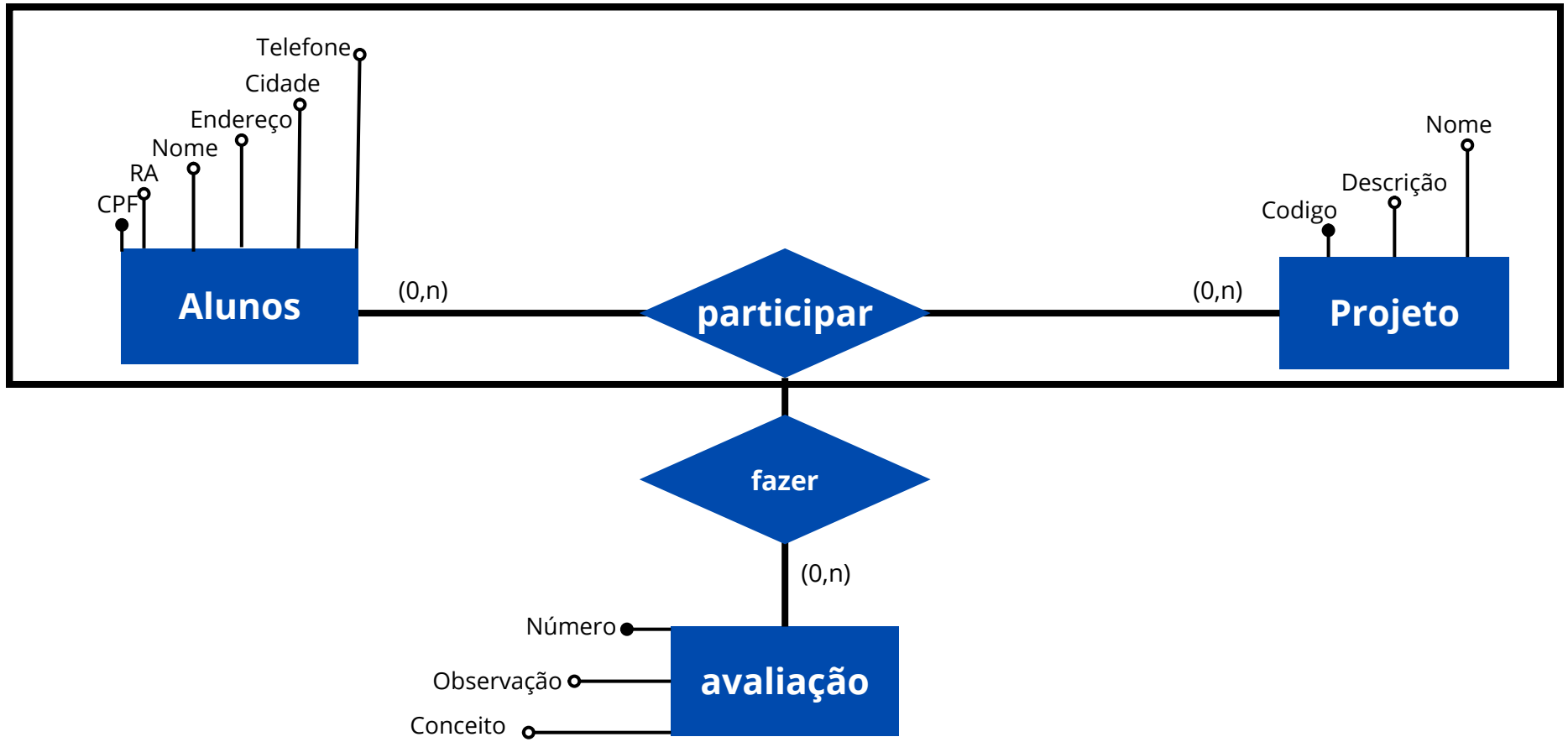
Opção 1

Opção 2



Agregação

Refere-se a uma forma de associar objetos ou entidades em um modelo, indicando que um objeto é composto por outros objetos. É uma maneira de representar a relação "todo-parte" entre as entidades. Por exemplo:



Agregação

Mapeamento:

1) Incluir as tabelas das entidades

Alunos					
CPF	**Nome**	**RA**	**Telefone**	**Cidade**	**Endereço**
111	Amanda	20241	(62)4665-4464	Goiania, GO	Rua sol, 123
222	Carlos	20242	(54)5464-8875	Ibá,RS	Av. Lua,456
333	Beatriz	20243	(93)1516-7896	Óbidos, PA	Rua das flores, 789
444	Jose	20244	(11)5149-879	Ibí, SP	Av. das Rosas, 101

Projeto		
Código	**Descrição**	**Nome**
C1	Em andamento	Projeto X
C2	Aguardando recurso	Projeto Y
C3	Em andamento	Projeto Z

Avaliação		
Número	**Observação**	**Conceito**
N01	Reavaliar	A
N02	Iniciar	B
N03	Aguardando	C

Agregação

Mapeamento:

2) Monta-se a tabela do relacionamento “participar”, que deve conter, como regra, a chave primária da tabela Alunos e da tabela Projetos. Na tabela Participar, a chave primária deve ser composta pelas duas chaves estrangeiras

Participar	
CPF_Alunos	**Código_Projeto**
111	C1
222	C1
222	C2
444	C3

Agregação

Mapeamento:

3) No entanto, surge uma dúvida, como compor a tabela do relacionamento “fazer”? Esse relacionamento está ligado a uma agregação e, assim, construímos a tabela Alunos-Projetos-Avaliações com as chaves primárias das três entidades, em que essas três chaves estrangeiras em conjunto formam a chave primária.

Fazer		
CPF_Alunos	**Código_Projeto**	**Número_Projeto**
111	C1	N02
222	C1	N03
222	C2	N04
444	C3	N04

Quando utilizar um banco de dados relacional?

O uso de banco de dados relacionais é recomendado em cenários que demandam estrutura organizada e consistente, com a linguagem SQL desempenhando papel essencial na manipulação e gerenciamento dos dados. Algumas situação que seja recomendado:

- Apropriado para dados com estrutura definida e relações claras entre entidades.
- Garantir a integridade de dados críticos em áreas como finanças, estoque e registros de pacientes.
- Se precisar de consultas complexas, junções, filtragem avançada, agrupamento e cálculos agregados, um banco de dados relacional é ideal.
- Deseja oferecer recursos avançados de segurança, como controle de acesso e criptografia, para garantir conformidade com normas e regulamentações.

UniSENAI

Modelo não relacional

Mundo real

- Com o crescimento e popularização da internet, as exigências cobradas de um banco de dados mudaram. Aplicações como portais de notícias exigem um alto volume de leitura por conta da audiência.
- Pense no banco de dados que armazena todos os mais de 1 bilhão de usuários ativos no Facebook, além de todo o conteúdo que eles geram diariamente.
- Além disso, algumas necessidades de negócio são muito complicadas de implementar em cima de um banco relacional.
- Baseado nessas novas necessidades e novos modelos de aplicações, um grupo de pessoas começou a ressaltar a importância na forma de armazenar os dados, e daí surgiu o movimento chamado NoSQL.

Modelo não relacional

Um modelo não relacional, também conhecido como NoSQL (Not Only SQL), é um tipo de banco de dados projetado para lidar com grandes volumes de dados de maneira flexível e escalável. Apresenta algumas características chaves como:

- Flexibilidade de esquema, permitindo armazenamento de dados sem um esquema rígido predefinido, facilitando a adaptação a mudanças nos requisitos de dados.
- Escalabilidade Horizontal nos bancos de dados NoSQL permite lidar com grandes volumes de dados e picos de tráfego distribuindo dados em vários servidores, mantendo o desempenho.
- Desnormalização de Dados em bancos de dados NoSQL visando otimizar a recuperação e desempenho, armazenando dados relacionados juntos para reduzir operações de junção.
- São ideais para armazenar e manipular dados como documentos JSON, pares chave-valor e grafos, sendo adequados para casos onde os dados não possuem estrutura fixa.
- São projetados para serem altamente disponíveis e tolerantes a falhas.

Modelo não relacional

- Os bancos de dados NoSQL usam uma variedade de modelos de dados para acessar e gerenciar os dados.

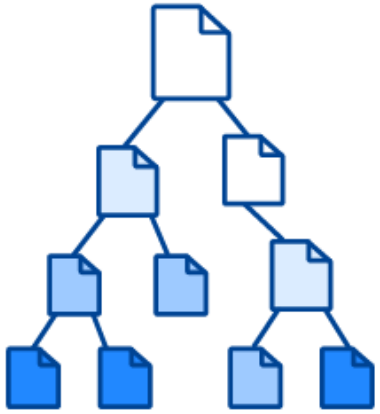
- Esses tipos de bancos de dados são otimizados especificamente para aplicações que exigem modelos de dados flexíveis, grande volume de dados e baixa latência, o que é conseguido relaxando algumas das restrições de consistência de dados dos bancos de dados relacionais.

- Há diferenças na implementação com base no modelo de dados. No entanto, muitos bancos de dados NoSQL usam Javascript Object Notation (JSON), um formato aberto de intercâmbio de dados que representa os dados como uma coleção de pares de nome-valor.

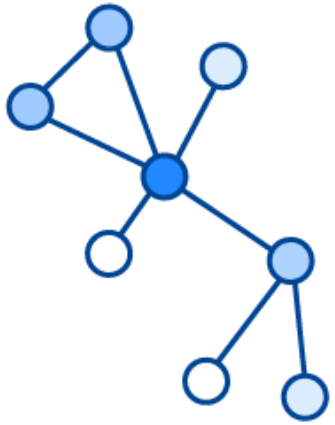
Modelo não relacional

Para esses bancos de dados NoSQL, temos uma variedade de modelos, incluindo:

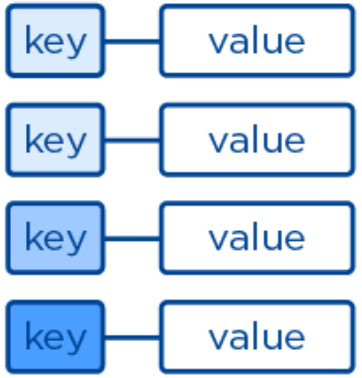
Modelo orientado a documentos



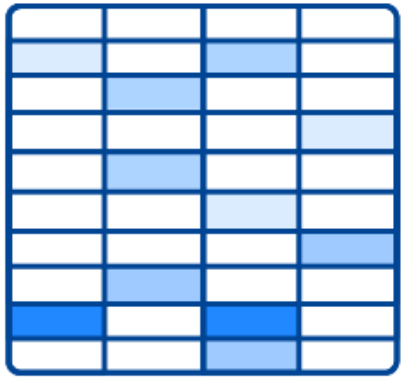
Modelo de grafos



Chave-valor



Modelo colunar



Modelo orientado a documentos

Nesse modelo, os dados são armazenados em documentos no formato JSON. Cada documento é identificado por uma chave única e pode conter diversas informações, como atributos e subdocumentos. Esse modelo é interessante para aplicações que exigem flexibilidade na estrutura dos dados e que lidam com grande volume de informações.

Exemplos de Bancos de Dados: MongoDB, Couchbase, CouchDB.

Uso Típico: Útil para aplicativos com dados variáveis ou semiestruturados, como redes sociais, sistemas de gerenciamento de conteúdo (CMS) e aplicativos web.

Na **Expedia**, a empresa utiliza o modelo flexível do **MongoDB** que facilita o armazenamento de qualquer combinação de cidades, datas e destinos.

Modelo de grafos

Neste modelo os dados são usados para armazenar dados interconectados, como em redes sociais ou sistemas de recomendação. Com o modelo de grafos, é possível fazer buscas detalhadas nas relações entre os dados, mesmo em bancos com centenas de milhares de relacionamentos.

Exemplos de Bancos de Dados: Neo4j, Amazon Neptune, ArangoDB.

Uso Típico: Útil para aplicativos que envolvem estruturas complexas de dados relacionais, como redes sociais, análise de redes, recomendações e sistemas de recomendação.

O **Medium**, por exemplo, utiliza o **Neo4j** para criar grafos que representam as conexões entre usuários e artigos, permitindo a montagem de um sistema de recomendação.

Chave-valor

Os dados são armazenados em pares de chave-valor, o que significa que cada dado é identificado por uma chave única. Esse modelo é ideal para aplicações que exigem alta performance em leitura e gravação de dados, como em aplicações de cache ou armazenamento de sessões de usuários.

Exemplos de Bancos de Dados: Redis, Amazon DynamoDB, Riak.

Uso Típico: Ideal para casos de uso que exigem acesso rápido e simples aos dados, como armazenamento de sessão, cache, contadores e sistemas de votação.

Este modelo usado pelo **Twitter**, que utiliza o **Redis** para implementar recursos em tempo real como contadores de retweets, curtidas e seguidores.

Modelo colunar

O banco de dados NoSQL de tipo colunar difere do modelo relacional por não utilizar tabelas. Os dados são organizados em colunas (ou linhas) próprias, facilitando a gestão rápida e simples de grandes conjuntos de dados,ou seja, todos os registros fazem parte da mesma tabela, mas cada um deles pode ter colunas diferentes. Algo essencial para aplicações de Big Data pois permite a busca e leitura apenas das colunas relevantes, economizando recursos de processamento.

Exemplos de Bancos de Dados: Apache Cassandra, HBase, ScyllaDB.

Uso Típico: Ideal para aplicativos que exigem acesso rápido e eficiente a grandes conjuntos de dados, como análise de big data e sistemas de gerenciamento de registros.

Uma das empresas que utilizam esse modelo é a **Netflix**, que utiliza o **Cassandra** para gravações de volume muito alto com baixa latência.

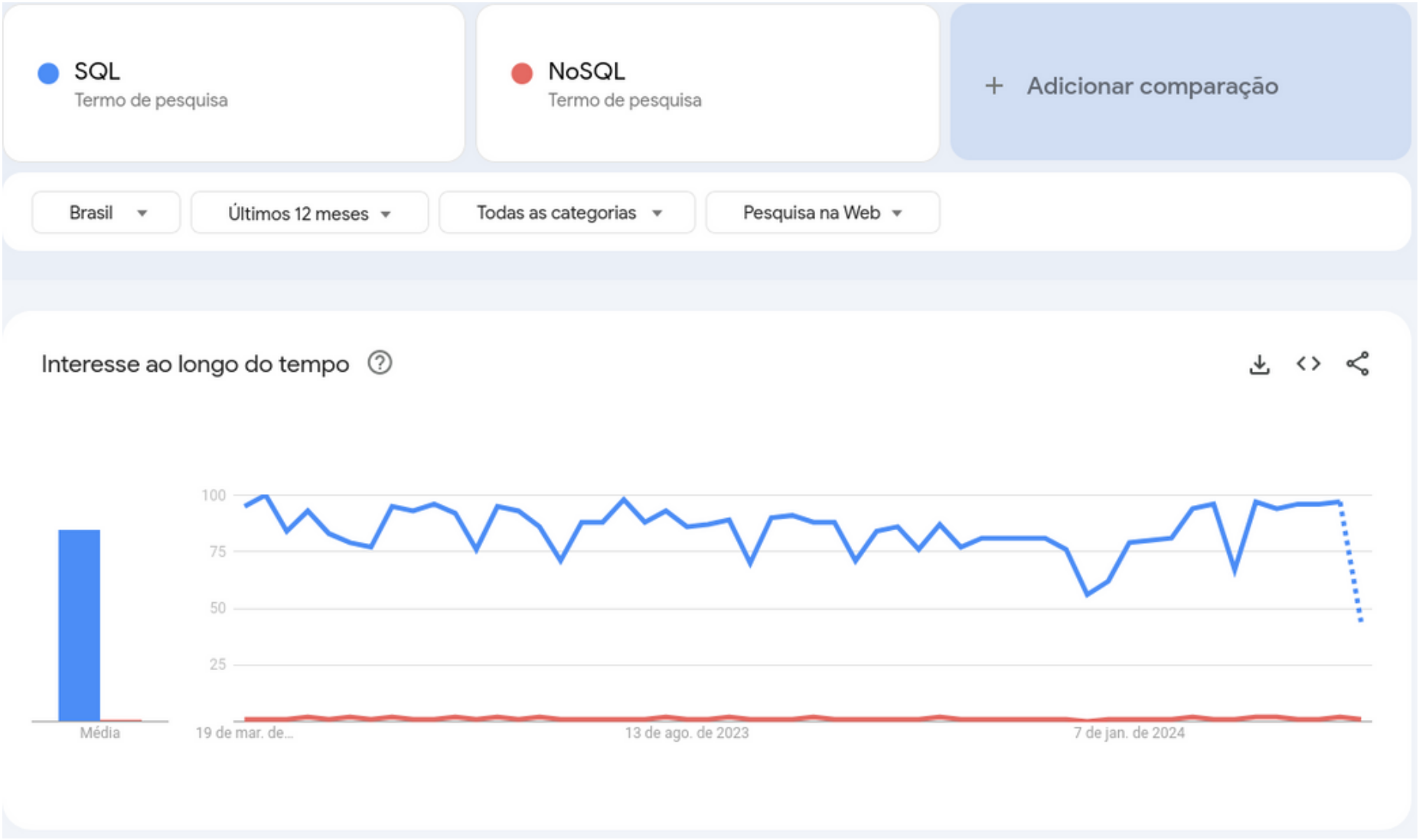
Quando você deve escolher bancos de dados NoSQL em vez de bancos de dados SQL?

Um banco de dados **NoSQL** é mais adequado para lidar com dados **indeterminados, não relacionados ou em constante mudança**. É fácil de usar para os desenvolvedores quando a aplicação define o esquema do banco de dados. Ele pode ser utilizado em aplicações que:

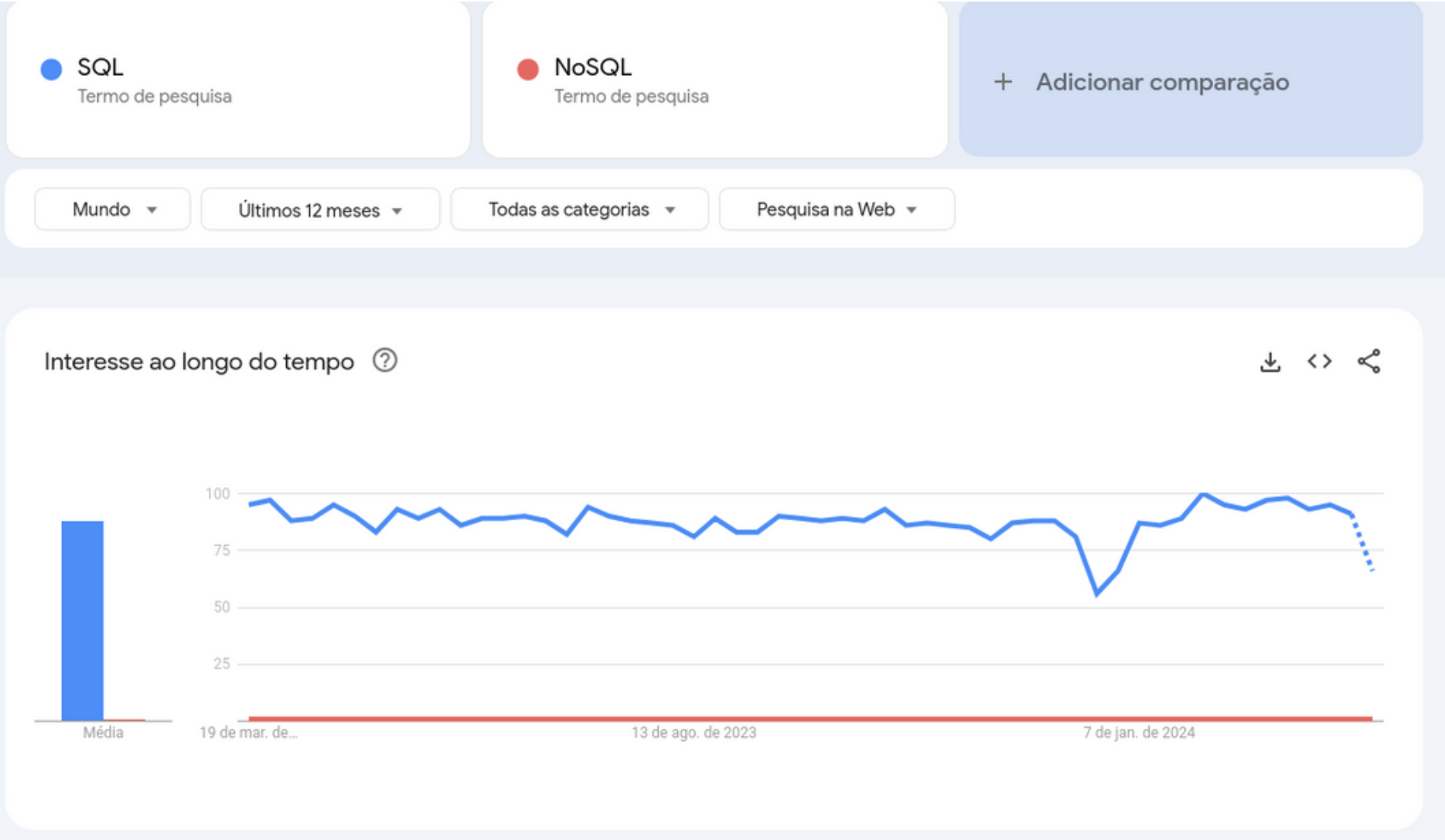
- Requerem esquemas flexíveis para permitir um desenvolvimento rápido e iterativo.
- Priorizam o desempenho em vez da consistência rígida dos dados e da manutenção de relacionamentos entre tabelas de dados (integridade referencial).
- Necessitam de escalabilidade horizontal, distribuindo-se por vários servidores.
- Suportam dados semiestruturados e não estruturados.

Não é necessário sempre escolher entre um banco de dados relacional e não relacional. **Uma mistura de bancos de dados SQL e NoSQL pode ser adotada em suas aplicações.** Essa abordagem híbrida é comum e garante que cada carga de trabalho seja direcionada para o banco de dados adequado, garantindo ótimo desempenho a um custo razoável.

NoSQL vs SQL


SQL
Termo de pesquisa
NoSQL
Termo de pesquisa
+ Adicionar comparação
Brasil
Últimos 12 meses
Todas as categorias
Pesquisa na Web
Interesse ao longo do tempo
100
75
50
25
Média
19 de mar. de...
13 de ago. de 2023
7 de jan. de 2024

NoSQL vs SQL


SQL
Termo de pesquisa
NoSQL
Termo de pesquisa
+ Adicionar comparação
Mundo
Últimos 12 meses
Todas as categorias
Pesquisa na Web
Interesse ao longo do tempo
100
75
50
25
Média
19 de mar. de...
13 de ago. de 2023
7 de jan. de 2024

UniSENAI

MONGO DB

Conhecendo o MongoDB

- O MongoDB é provavelmente o banco NoSQL **mais usado** no mundo atualmente.
- Se enquadra na categoria dos bancos com armazenamento baseado em documentos.
- No caso do MongoDB, quando persistimos um documento em uma coleção, o equivalente a uma linha em uma tabela, os dados ficam armazenados em um formato muito semelhante ao JSON, chamado de BSON (http://bsonspec.org/).
- Assim como as tabelas possuem colunas, um documento possui seus campos.
- Além da terminologia, outro ponto que bancos orientados a documentos diferem dos relacionais é que cada documento de uma coleção pode ter qualquer **campo**. Ou seja, **não existe o que é conhecido nos bancos relacionais por esquema.**

UniSENAI

Conhecendo o MongoDB

Termos/conceitos do SQL	Termos/conceitos do MongoDB
Database	Database
Tabela	Coleção
Linha	Documento ou documento BSON
Coluna	Campo
Index	Index
Table join	Documentos aninhado (*embedded*) e vinculados
Chave primária — especifica qualquer coluna única ou uma combinação de colunas como chave primária	Chave primária — No MongoDB, a chave primária é automaticamente definida como campo *_id*
Agregação (*group by*)	Agregação de pipeline

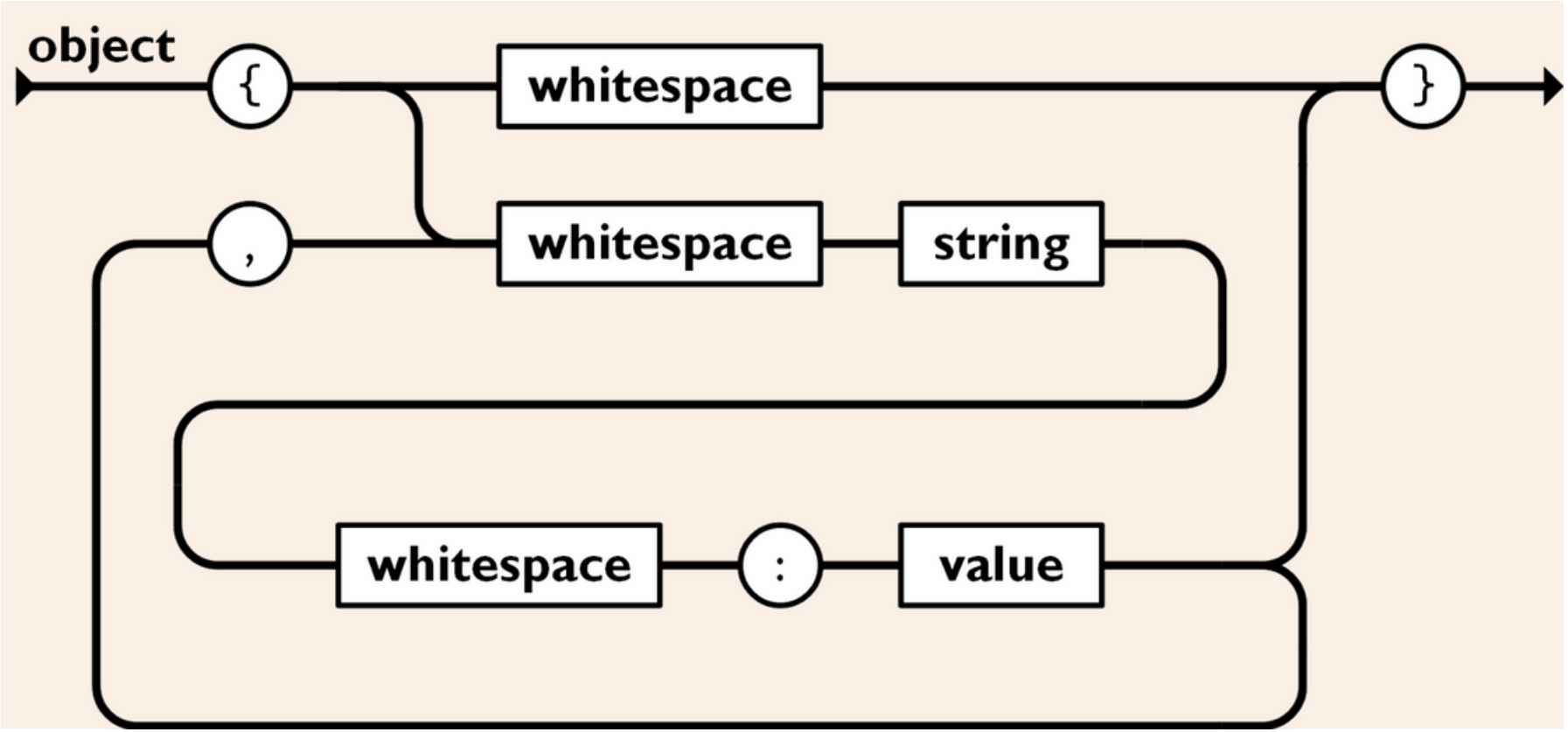
MongoDB Community e MongoDB Compass

- O MongoDB Community é um **banco de dados** NoSQL de código aberto altamente escalável e flexível, projetado para armazenar e recuperar dados de forma eficiente, com suporte a replicação e alta disponibilidade.

- O MongoDB Compass é **sistema de gerenciamento de banco de dados (SGBD)**, ou seja, uma GUI intuitiva que simplifica o desenvolvimento, administração e monitoramento de bancos de dados MongoDB, oferecendo visualização de dados, criação de consultas e gerenciamento de índices de forma amigável e eficiente.

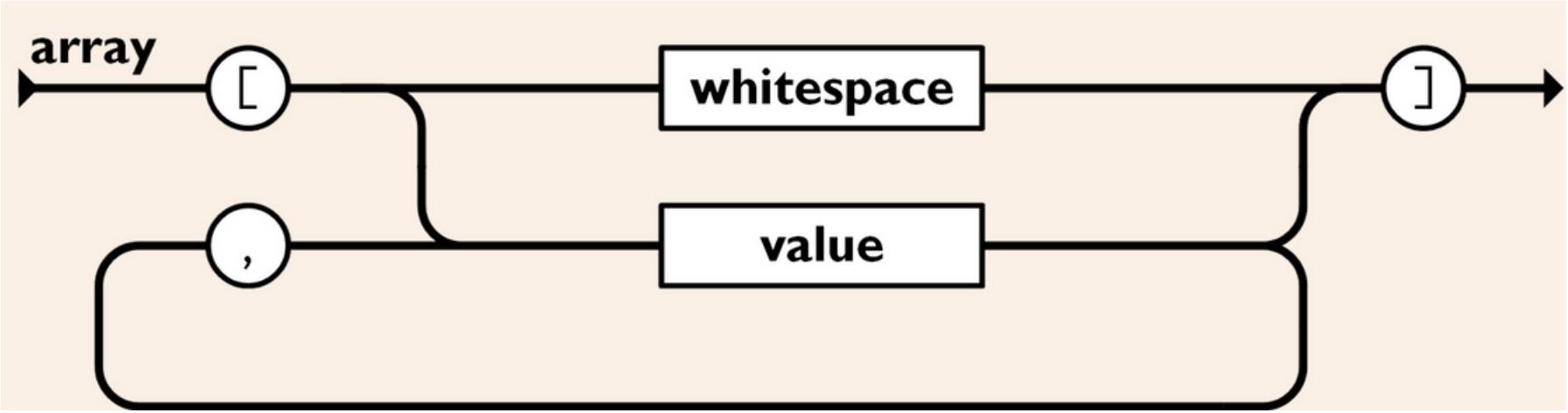
Dados em Json

Um objeto é um conjunto desordenado de pares nome/valor. Um objeto começa com "**{" chave de abertura** e termina com **"}" chave de fechamento.** Cada nome é seguido por **":"dois pontos** e os pares nome/valor são seguidos por **","vírgula.**



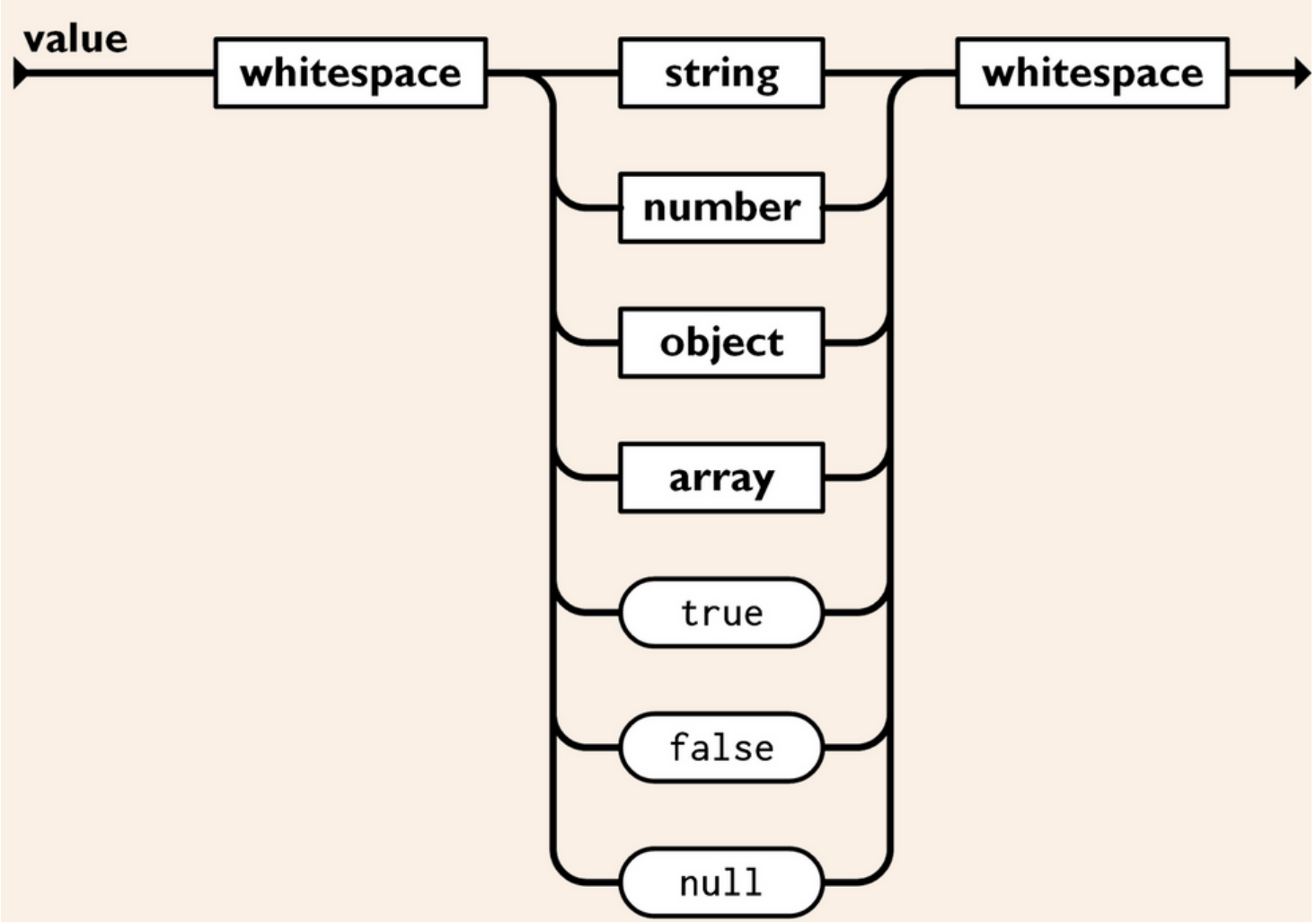
Dados em Json

Uma array é uma coleção de valores ordenados. O array começa com **“[”conchete de abertura** e termina com **“]”conchete de fechamento**. Os valores são separados por “**,”vírgula**.



Dados em Json

Um valor (value, na imagem acima) pode ser uma cadeia de caracteres (string), ou um número, ou true ou false, ou null, ou um objeto ou uma array. Estas estruturas podem estar aninhadas.



Exemplo cliente


```
{
 "_id": 1,
 "nome": "João Silva",
 "endereco": "Rua Principal, 123",
 "cidade": "São Paulo",
 "email": "joao@example.com",
 "historico_compras": [
  {
   "id_pedido": 101,
   "data_compra": "2024-03-25",
   "total": 50.00,
   "itens": [
    {
     "produto": "Camiseta",
     "quantidade": 1,
     "preco_unitario": 25.00
    }
   ]
  }
 ]
}
```

Exemplo cliente


```
_id: 1
nome : "João Silva"
endereco : "Rua Principal, 123"
cidade : "São Paulo"
email : "joao@example.com"
▾ historico_compras : Array (1)
  ▾ 0: Object
      id_pedido : 101
      data_compra : "2024-03-25"
      total : 50
    ▾ itens : Array (1)
      ▾ 0: Object
          produto : "Camiseta"
          quantidade : 1
          preco_unitario : 25
```

Exemplo cliente

Cliente 2 - Maria Oliveira:

Nome: Maria Oliveira
Endereço: Avenida Central, 456
Cidade: Rio de Janeiro
E-mail: maria@example.com
Histórico de Compras:
Pedido ID: 102
Data da Compra: 2024-03-24
Total: R$ 75.00
Itens:
Produto: Sapato
Quantidade: 1
Preço Unitário: R$ 50.00
Produto: Bolsa
Quantidade: 1
Preço Unitário: R$ 25.00

Cliente 3 - Carlos Santos:

Nome: Carlos Santos
Endereço: Rua das Flores, 789
Cidade: Belo Horizonte
E-mail: carlos@example.com
Histórico de Compras:
Pedido ID: 103
Data da Compra: 2024-03-23
Total: R$ 120.00
Itens:
Produto: Jaqueta
Quantidade: 1
Preço Unitário: R$ 80.00
Produto: Camisa
Quantidade: 2
Preço Unitário: R$ 20.00
Pedido ID: 104
Data da Compra: 2024-03-25
Total: R$ 60.00
Itens:
Produto: Calça Jeans
Quantidade: 1
Preço Unitário: R$ 60.00

Exemplo cliente

Também tem a opção de importar arquivos em Json.

Observação: O arquivo já esta no AVA.

Cliente 4 e 5 só tem uma pessoa por arquivo, já cliente 6_7 são dois. Observe como esta o formato.

Exemplo cliente - Operações

Inserção de Documentos:

Esta operação insere um novo documento na coleção "clientes" do banco de dados. O método insertOne() é utilizado para inserir um único documento na coleção especificada.

```
db.clientes.insertOne({
    "_id": 8,
    "nome": "Lucas Fernandes",
    "endereco": "Rua das Árvores, 456",
    "cidade": "Recife",
    "email": "lucas@example.com",
    "historico_compras": []
})
```

Exemplo cliente - Operações

Atualização de Documentos:

Esta operação atualiza um documento existente na coleção "clientes". O método updateOne() é utilizado para atualizar o primeiro documento que corresponde ao filtro especificado.

```
db.clientes.updateOne(
    { "_id": 8 },
    { $set: { "cidade": "Olinda" } }
)
```

Exemplo cliente - Operações

Consulta de Documentos:

Esta operação consulta um documento na coleção "clientes" com base em um critério de pesquisa. O método findOne() é utilizado para encontrar o primeiro documento que corresponde ao filtro especificado.

db.clientes.**findOne**({ "nome": "Lucas Fernandes" })

Exemplo cliente - Operações

Remoção de Documentos:

Esta operação remove um documento da coleção "clientes". O método deleteOne() é utilizado para remover o primeiro documento que corresponde ao filtro especificado.

db.clientes.**deleteOne**({ "_id": 8 })

Exemplo cliente - Operações

Consulta com Projeção:

Esta operação consulta documentos na coleção "clientes" com base em um critério de pesquisa e projeta apenas os campos "nome" e "endereco" para cada documento retornado. Isso ajuda a reduzir a quantidade de dados transferidos entre o servidor e o cliente.

db.clientes.**find**({ "cidade": "Recife" }, { "nome": 1, "endereco": 1 })

Além disso pode utilizar o comando sort() é um método que permite ordenar os documentos com base em um campo específico, usando 1 para ordem crescente e -1 para ordem decrescente.

Exemplo cliente - Operações

Ordenação em Ordem Alfabética:

Esta operação encontra todos os documentos na coleção "clientes" e os ordena em ordem alfabética crescente com base no campo "nome". O número 1 indica que a ordenação será feita em ordem ascendente. Para ordenação em ordem decrescente, você pode usar -1 em vez de 1.

```
db.clientes.find().sort({ "nome": 1 })
```

Exemplo cliente - Operações

Agregação de Documentos:

Esta operação de agregação começa filtrando os documentos da coleção "clientes" onde a cidade é "Recife" usando o estágio $match. Em seguida, no estágio $group, os documentos filtrados são agrupados pelo campo "cidade" e é calculado o total de clientes em cada cidade usando o operador $sum. O resultado será o número total de clientes na cidade de Recife.

```
db.clientes.aggregate([
    { $match: { "cidade": "Recife" } },
    { $group: { _id: "$cidade", total_clientes: { $sum: 1 } } }
])
```

UniSENAI

Filmes e programas

Agora é com vocês.

Vocês irão entrar no AVA e inserir o arquivo "filmes.json" que está no tópico aula 5 no MongoDB Compass. Com esses dados, vocês irão realizar as seguintes tarefas:

1. Encontrar quais filmes são de comédia.
2. Encontrar quais filmes têm nota de avaliação maiores que 8.
3. Encontrar o filme de Animação com a maior nota de avaliação.
4. Excluir todos os filmes com nota menor que 6.
5. Inserir 3 filmes que vocês gostam no banco de dados.


UniSENAI
94%
unisenaisc.com.br
SENAI