ESPECIALIZAÇÃO EM INTERNET DAS COISAS

Apresentação de Seminário – Disciplina Banco de Dados

17/06/2025

INSTITUTO FEDERAL
Paraná
Campus
Foz do Iguaçu

Comparativo entre Banco de Dados Relacional (PostgreSQL) e Não Relacional (MongoDB)

Estudo de Caso: Casos de COVID-19 no Brasil

Linguagem de programação aplicada: Python

Aluno: Heber Miguel dos Santos
heber.eng.eletronica@gmail.com

SUMÁRIO

INSTITUTO FEDERAL
Paraná
Campus
Foz do Iguaçu

INSTITUTO FEDERAL
Paraná
Campus
Foz do Iguaçu

INTRODUÇÃO E CONTEXTUALIZAÇÃO

INTRODUÇÃO E CONTEXTUALIZAÇÃO

INSTITUTO FEDERAL
Paraná
Campus
Foz do Iguaçu

Objetivo da Apresentação:

- Apresentar diferenças entre os paradigmas Relacional e Não Relacional.
- Comparar PostgreSQL (SQL) e MongoDB (NoSQL).
  - Armazenamento de dados;
  - Consultas;
- Demonstrar consultas usando dados reais da COVID-19 no Brasil.

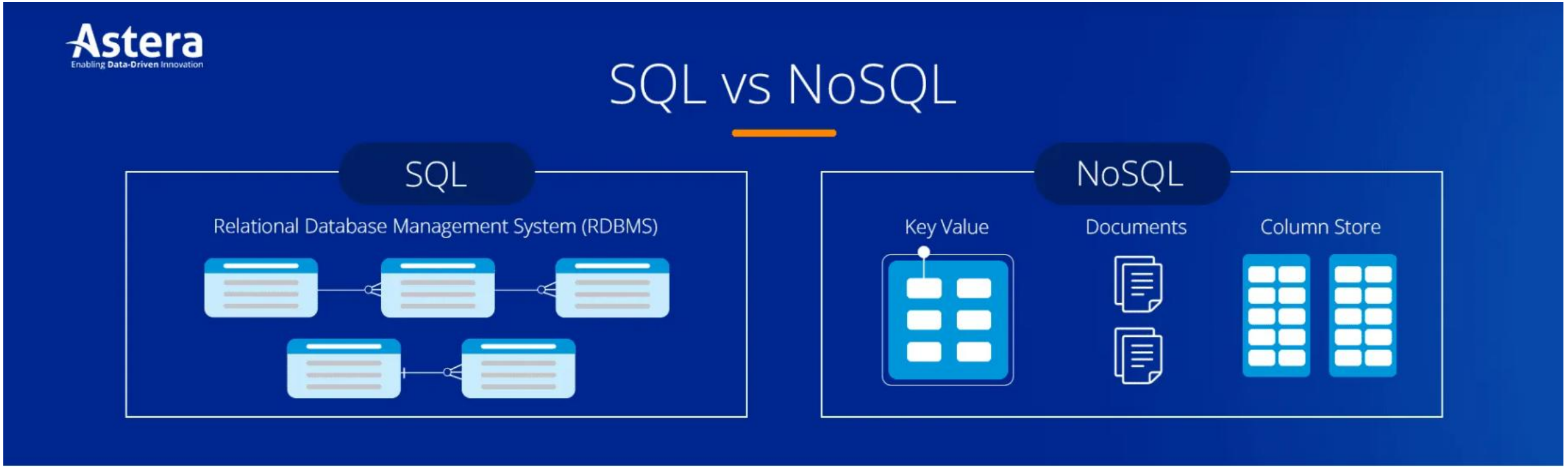
Características do Dataset:

- Repositório de dados públicos disponibilizados em formato CSV.
  - Projeto Brasil.io
  - URL: COVID-19 - Datasets - Brasil.IO
- Fonte: CSV com registros por estado e cidade. Aproximadamente
- Total de registros: 3,9mi de registros.
- Atributos: cidade, data, estado, casos confirmados, mortes, população, etc.
- Total de 18 atributos

Atributos do DATASET					
city	city_ibge_code	date	epidemiological_week	estimated_population	estimated_population_2019
is_last	is_repeated	last_available_confirmed	last_available_confirmed_per_100k_inhabitants	last_available_date	last_available_death_rate
last_available_deaths	order_for_place	place_type	state	new_confirmed	new_deaths

INSTITUTO FEDERAL
Paraná
Campus
Foz do Iguaçu

Os paradigmas de banco dados



Fonte: SQL vs. NoSQL: Diferenças, vantagens e casos de uso | Astera

INSTITUTO FEDERAL
Paraná
Campus
Foz do Iguaçu

COMPARATIVOS ENTRE OS DOIS TIPOS DE BANCOS DE DADOS

COMPARATIVOS ENTRE OS DOIS TIPOS DE BANCOS DE DADOS

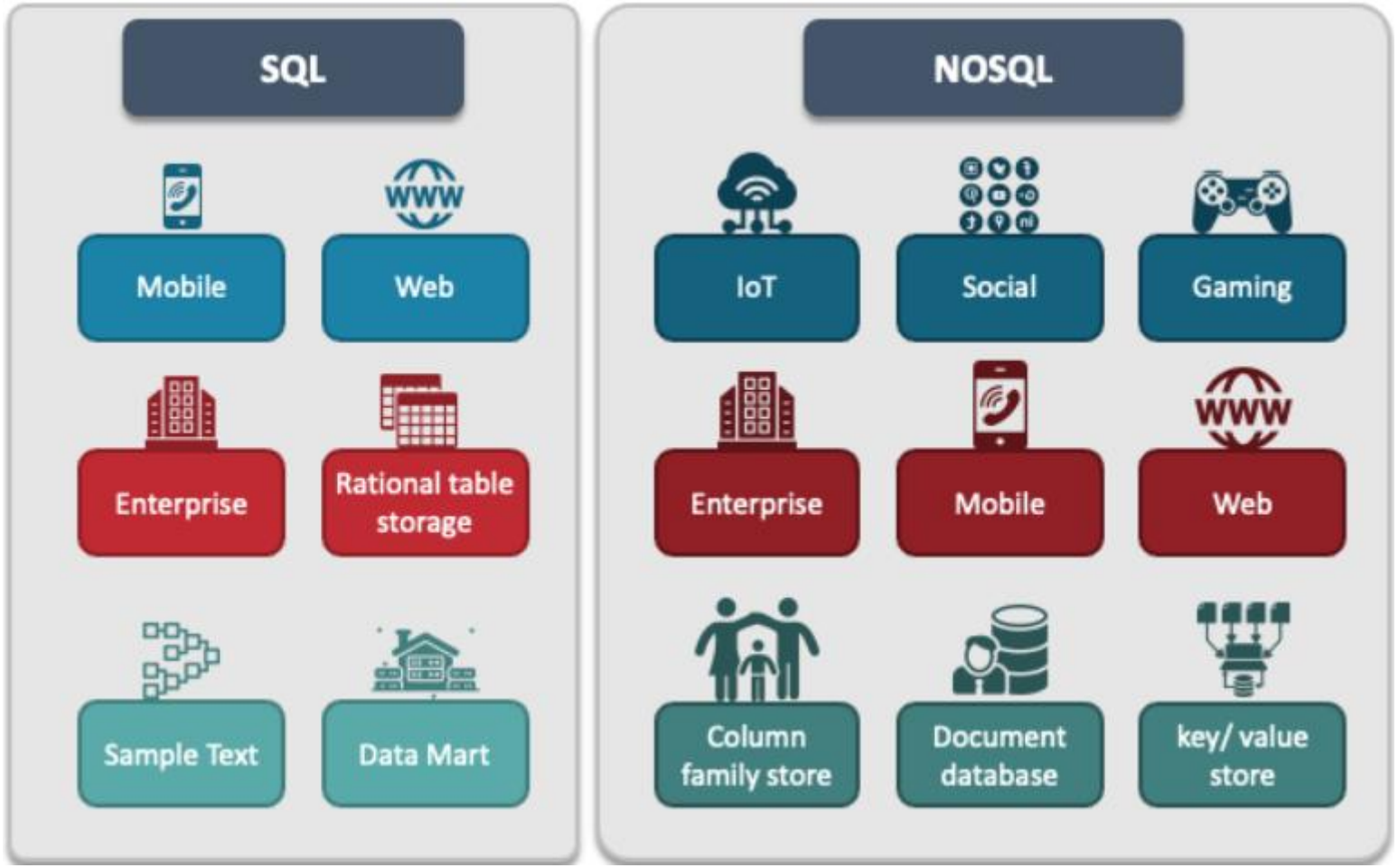
INSTITUTO FEDERAL
Paraná
Campus
Foz do Iguaçu

Principais diferenças:

Característica	SQL	NoSQL
Modelo de dados	Relacional (Tabelas)	Não relacional (chave-valor, documento, família de colunas, gráfico)
Esquema	Esquema fixo com estrutura predefinida	Esquema dinâmico, estrutura flexível
Escalabilidade	Escala vertical (adicionando recursos a um único servidor)	Escala horizontal (distribuição em vários nós)
Tratamento de transações	Segue propriedades ACID para confiabilidade	Segue o teorema CAP, geralmente priorizando a disponibilidade e a tolerância à partição
Desempenho	Otimizado para consultas e transações complexas	Otimizado para armazenamento de dados em larga escala e análises em tempo real
Linguagem de consulta	Utiliza SQL (Linguagem de Consulta Estruturada)	Varia de acordo com o banco de dados (por exemplo, consultas do tipo JSON no MongoDB, CQL no Cassandra)
Flexibilidade	Estrutura rígida com relacionamentos	Sem esquema ou semiestruturado para adaptabilidade
Comunidade e Suporte	Grande e bem estabelecida comunidade e suporte empresarial	Comunidades de código aberto em crescimento com soluções apoiadas por fornecedores
Casos de uso	Sistemas financeiros, ERP, CRM	Aplicações de big data, IoT, análises em tempo real

Fonte: SQL vs. NoSQL: Diferenças, vantagens e casos de uso | Astera

Principais diferenças de Aplicações:



Fonte: https://www.sketchbubble.com/en/presentation-sql-vs-nosql.html

INSTITUTO FEDERAL
Paraná
Campus
Foz do Iguaçu

BANCO DE DADOS RELACIONAL (SQL)

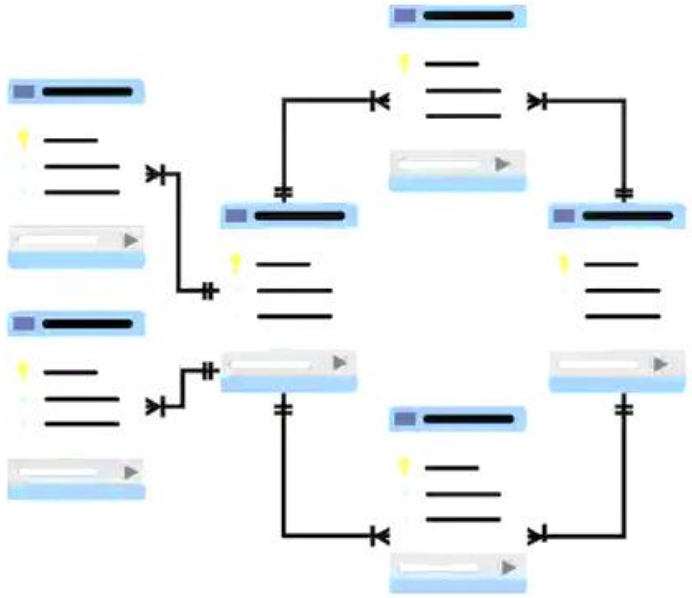
BANCO DE DADOS RELACIONAL

- Modelo baseado em tabelas com colunas fixas.
- Usa Linguagem SQL (Structured Query Language).
- Garantia de integridade, ACID (Atomicidade, Consistência, Isolamento, Durabilidade).
- Suporta JOINs e normalização.
- Exemplo: PostgreSQL.

Exemplo:

PostgreSQL

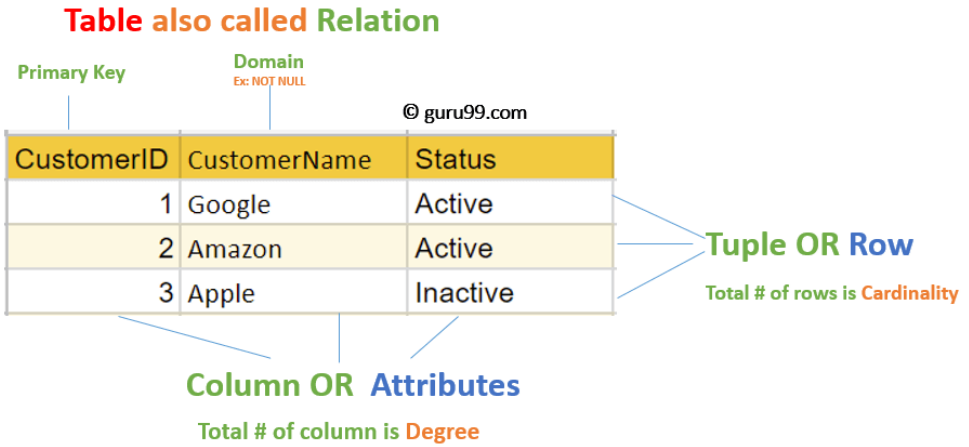
Relacional:



SQL Query:

```
SELECT DISTINCT
    Table1.*,
    Table2.*
FROM Table1
JOIN Table2
    ON matching_condition
WHERE constraint_expression
GROUP BY [columns]
HAVING constraint_expression
ORDER BY [columns]
LIMIT count
```

Tabelas:



Fontes:
[Relational Data Model in DBMS | Database Concepts & Example]

INSTITUTO FEDERAL
Paraná
Campus
Foz do Iguaçu

BANCO DE DADOS NÃO RELACIONAL (NoSQL)

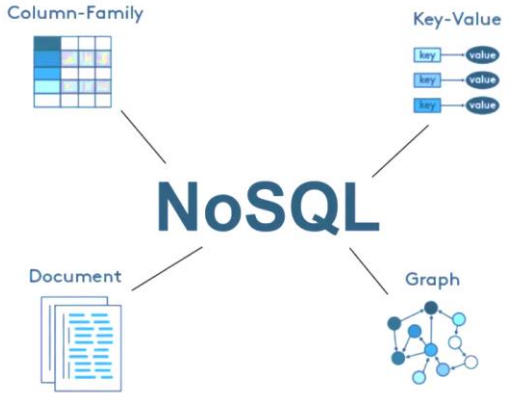
BANCO DE DADOS NÃO RELACIONAL (NoSQL)

- Modelo baseado em documentos (JSON), grafos, chave-valor, colunas largas.
- Sem linguagem definida;
- Estrutura flexível e schema-less.
- Ideal para alta escalabilidade horizontal e dados sem estrutura fixa.
- Construção das consultas complexas usando agregações e pipelines.
- Exemplo: MongoDB.

Exemplo:

mongoDB.

Documentos e Flexível:



JSON:

```
{
    "books": [
        {
         "id": "01",
         "language": "Java",
         "author": "H. Javeson",
         "year": 2015
        },
        {
         "id": "07",
         "language": "C++",
         "edition": "second",
         "author": "E. Sepp",
         "price": 22.25
        },
    ]
}
```

NoSQL Query:

```
db.users.find(
    { age: { $gt: 18 } },
    { name: 1, address: 1 }
).limit(5)
```

Fontes: https://www.meilhaus.de/en/rtsa-suite-pro-122-035.htm

INSTITUTO FEDERAL
Paraná
Campus
Foz do Iguaçu

COMPARATIVO DE MANIPULAÇÕES

COMPARATIVO DE MANIPULAÇÕES

INSTITUTO FEDERAL
Paraná
Campus
Foz do Iguaçu

CRUD de cada tipo de banco de dados:

Resumo		
Operação	**MongoDB (NoSQL)**	**PostgreSQL (SQL)**
Create	`insertOne({...})`	`INSERT INTO ... VALUES (...)`
Read	`find({filtro}, {projeção})`	`SELECT campos FROM ... WHERE ...`
Update	`updateMany({filtro}, {$set})`	`UPDATE ... SET ... WHERE ...`
Delete	`deleteMany({filtro})`	`DELETE FROM ... WHERE ...`

NoSQL Query:

```
db.users.find(                          <-- collection
   { age: { $gt: 18 } },                <-- query criteria
   { name: 1, address: 1 }              <-- projection
).limit(5)                              <-- cursor modifier
```

SQL Query:

```
SELECT _id, name, address   <-- projection
FROM   users                <-- table
WHERE  age > 18             <-- select criteria
LIMIT  5                    <-- cursor modifier
```

Fontes:
Adaptado de: https://devopedia.org/mongodb-query-language

SIMPLES EXEMPLOS - CONEXÃO

INSTITUTO FEDERAL
Paraná
Campus
Foz do Iguaçu

MongoDB

```
from pymongo import MongoClient
client = MongoClient("mongodb://localhost:27017/")
db = client["meu_banco"]
users = db["users"]
```

PostgreSQL

```
import psycopg2
conn = psycopg2.connect(dbname="meu_banco", user="usuario",
password="senha", host="localhost")
cur = conn.cursor()
```

Fontes:
Adaptado de: https://devopedia.org/mongodb-query-language

INSERÇÃO – (INSERT)

INSTITUTO FEDERAL
Paraná
Campus
Foz do Iguaçu

MongoDB

```
########## CREATE - Inserir dados ###################
users.insert_one({
    "name": "sue",
    "age": 26,
    "status": "pending"
})
```

Representação:

```
{
  "_id": ObjectId("xxx"),
  "name": "sue",
  "age": 26,
  "status": "pending"
}
```

PostgreSQL

```
########## CREATE - Inserir dados ###################
cur.execute(
    "INSERT INTO users (name, age, status) VALUES (%s, %s, %s);",
    ("sue", 26, "pending")
)
```

Representação:

```
Tabela: users

+----+------+-----+---------+
| id | name | age | status  |
+----+------+-----+---------+
| 1  | sue  | 26  | pending |
+----+------+-----+---------+
```

Fontes:
Adaptado de: https://devopedia.org/mongodb-query-language

LEITURA – (READ)

INSTITUTO FEDERAL
Paraná
Campus
Foz do Iguaçu

MongoDB

```
########## READ – Consultar dados ###################
result = users.find(
    {"age": {"$gt": 18}},
    {"name": 1, "address": 1}
).limit(5)

for doc in result:
    print(doc)
```

PostgreSQL

```
########## READ – Consultar dados ###################
cur = conn.cursor()
cur.execute(
    "SELECT name, address FROM users WHERE age > %s LIMIT 5;",
    (18,)
)
for row in cur.fetchall():
    print(row)
```

Fontes:
Adaptado de: https://devopedia.org/mongodb-query-language

ATUALIZAÇÕES – (UPDATE)

INSTITUTO FEDERAL
Paraná
Campus
Foz do Iguaçu

MongoDB

```
########## UPDATE – Atualizar dados #################
users.update_many(
    {"age": {"$lt": 18}},
    {"$set": {"status": "reject"}}
)
```

PostgreSQL

```
########## UPDATE – Atualizar dados #################
cur = conn.cursor()
cur.execute(
    "UPDATE users SET status = %s WHERE age < %s;",
    ("reject", 18)
)
```

Fontes:
Adaptado de: https://devopedia.org/mongodb-query-language

REMOVER – (DELETE)

INSTITUTO FEDERAL
Paraná
Campus
Foz do Iguaçu

MongoDB

```
########## DELETE - Remover dados ###################
users.delete_many({"status": "reject"})
```

PostgreSQL

```
########## DELETE - Remover dados ###################
cur.execute(
    "DELETE FROM users WHERE status = %s;",
    ("reject",)
)
```

Fontes:
Adaptado de: https://devopedia.org/mongodb-query-language

INSTITUTO FEDERAL
Paraná
Campus
Foz do Iguaçu

MANIPULAÇÕES DO BANCO DE DADOS PostgreSQL

&

MANIPULAÇÕES DO BANCO DE DADOS MongoDB (NoSQL)

Vamos praticar com o caso de estudo...

INSTITUTO FEDERAL
Paraná
Campus
Foz do Iguaçu

CONCLUSÃO

CONCLUSÃO

INSTITUTO FEDERAL
Paraná
Campus
Foz do Iguaçu

PostgreSQL é ideal para:

- Aplicações com dados estruturados e relacionais.
- Sistemas que exigem integridade e consistência.

MongoDB é ideal para:

- Dados semi-estruturados e alta flexibilidade.
- Prototipação rápida e sistemas escaláveis.

Escolha depende do problema, não existe "melhor" absoluto.

CONCLUSÃO

Critérios	Escolha SQL se	Escolha NoSQL se
Estrutura de dados	Você precisa de um formato estruturado e tabular com esquema predefinido	Você precisa de um esquema flexível ou armazenar dados não estruturados/semiestruturados
Escalabilidade	O dimensionamento vertical (adicionar recursos a um único servidor) é suficiente	Você precisa de dimensionamento horizontal (adicionando mais servidores) para tráfego alto
Tratamento de transações	A conformidade com o ACID é crítica (por exemplo, transações financeiras)	A consistência eventual é aceitável e a alta disponibilidade é uma prioridade
Consultando Necessidades	Consultas SQL complexas, junções e agregações são necessárias	Pesquisas simples de valor-chave ou consultas distribuídas são mais comuns
Requisitos de desempenho	Transações e consistência são mais importantes que velocidade	Velocidade e processamento em tempo real são as principais prioridades
Casos de uso	Bancos, ERP, CRM, aplicativos empresariais tradicionais	Big data, IoT, mídia social, gerenciamento de conteúdo, análise em tempo real

Fonte: SQL vs. NoSQL: Diferenças, vantagens e casos de uso | Astera

CONCLUSÃO

INSTITUTO FEDERAL
Paraná
Campus
Foz do Iguaçu

Dicas Finais

- Estude o modelo de dados antes de escolher o banco.
- Combine ambos quando necessário (arquiteturas híbridas).
- Documentação e comunidade são grandes aliadas.

Fonte: SQL vs. NoSQL: Diferenças, vantagens e casos de uso | Astera

CONCLUSÃO

INSTITUTO FEDERAL
Paraná
Campus
Foz do Iguaçu

Bibliografia
“Links importantes e Materiais complementares”

SQL vs. NoSQL: Diferenças, vantagens e casos de uso | Astera
How to Use PostgreSQL in Python
Bancos De Dados NoSQL Versus SQL | MongoDB
PostgreSQL: The world's most advanced open source database
Download MongoDB Community Server | MongoDB
Instale o MongoDB Community Edition no Windows - Manual do banco de dados v8.0 - MongoDB Docs
O que é um banco de dados? – Explicação sobre bancos de dados na nuvem – AWS
What Is a Database? | Oracle
SQL vs. NoSQL Databases: What's the Difference? | IBM