What is **Cassandra?**

Introdução

O Cassandra é um banco de dados NoSQL altamente escalável e distribuído, projetado para lidar com grandes volumes de dados e garantir alta disponibilidade.

Ele foi desenvolvido inicialmente pelo Facebook e posteriormente doado à Apache Software Foundation, onde se tornou um projeto de código aberto.

Os desenvolvedores do Facebook nomearam seu banco de dados **em homenagem ao profeta mitológico Trojan Cassandra**, com alusões clássicas à maldição de um oráculo.

Características

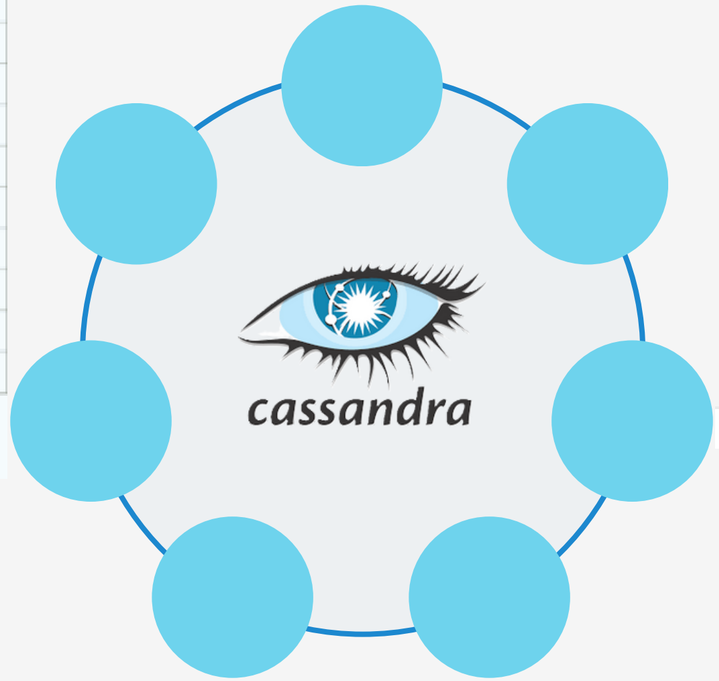
- **Arquitetura Descentralizada:** Ele foi projetado para operar em um **cluster de vários nós**, onde os dados são distribuídos de forma uniforme entre os nós. Essa **abordagem distribuída permite que o Cassandra forneça alta escalabilidade horizontal**, ou seja, a capacidade de aumentar a capacidade de armazenamento e processamento adicionando mais nós ao cluster.
- **Baseado em Colunas:** Isso significa que os dados são armazenados em colunas, em vez de linhas, o que permite uma flexibilidade maior na forma como os dados são organizados e recuperados
- **Linguagem CQL:** Possui uma linguagem de consulta própria, chamada Cassandra Query Language (CQL), que é baseada no SQL, facilitando a adoção por desenvolvedores familiarizados com bancos de dados relacionais.
- **O único campo obrigatório é a chave:** No Cassandra o único campo obrigatório é a chave, desse modo, podem existir registros com 10, 20, 100, ou nenhuma coluna, desde que o mesmo possua uma chave. Os campos são criados por demanda, por exemplo, se um registro não tiver o campo “telefone” o mesmo não existirá, diferentemente de um banco relacional, em que a coluna existe para todos os registros com o valor null.

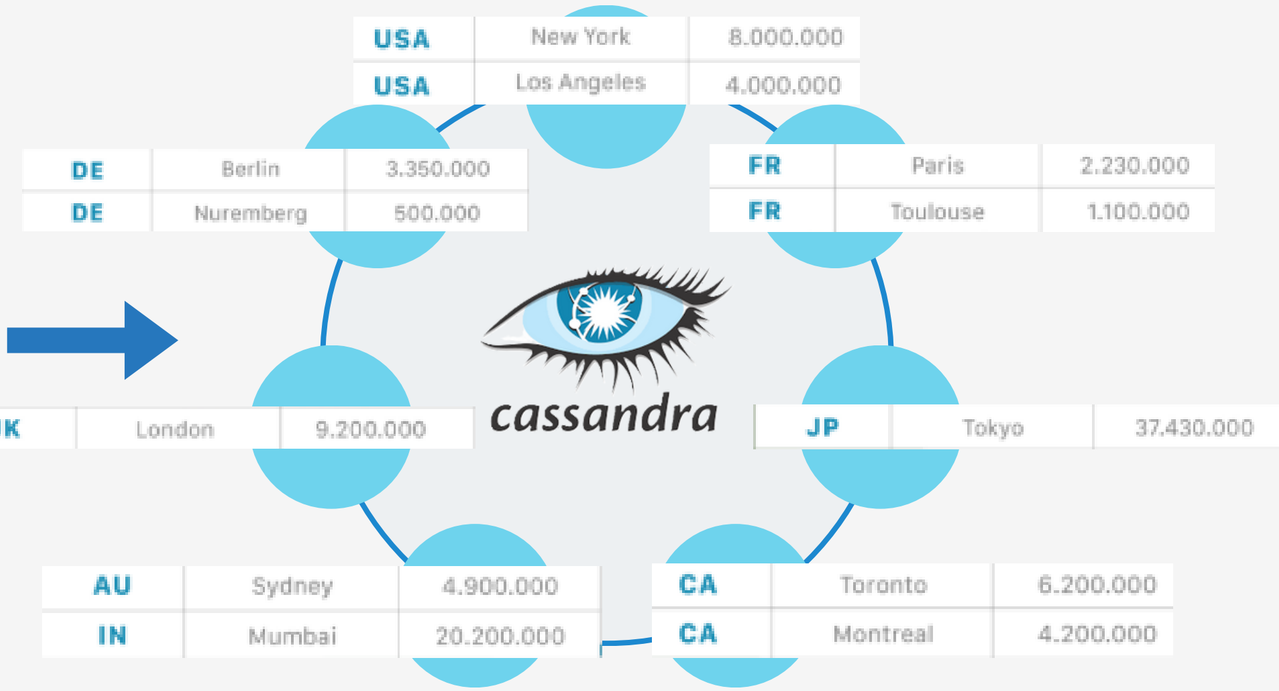
Arquitetura do Cassandra

Os dados são armazenados nos nós, e todos os registros com a mesma chave são armazenados no mesmo nó. Isso torna as consultas mais rápidas do que em bancos de dados SQL, onde várias tabelas podem estar sendo executadas em várias máquinas.

COUNTRY	CITY	POPULATION
USA	New York	8.000.000
USA	Los Angeles	4.000.000
FR	Paris	2.230.000
DE	Berlin	3.350.000
UK	London	9.200.000
AU	Sydney	4.900.000
DE	Nuremberg	500.000
CA	Toronto	6.200.000
CA	Montreal	4.200.000
FR	Toulouse	1.100.000
JP	Tokyo	37.430.000
IN	Mumbai	20.200.000

Partition Key





Modelo de Dados: Cassandra vs Bancos Relacionais

Recurso	Cassandra	RDBMS
Organização	Keyspace → Tabela → Linha	Banco de dados → Tabela → Linha
Estrutura de linha	Colunas dinâmicas	Esquema fixo
Dados da coluna	Nome, tipo, valor, registro de data e hora	Nome, tipo, valor
Mudanças de esquema	Modificações de tempo de execução	Geralmente requer tempo de inatividade
Modelo de Dados	Desnormalizado	Normalizado com JOINs

Para que é usado o Cassandra?

Para que é usado o Cassandra?

- **Análise em tempo real:** O Cassandra fornece a infraestrutura necessária para lidar com o processamento e a análise de dados em larga escala para cargas de trabalho contínuas e de alta velocidade.
- **Gerenciamento de dados de IoT:** Como os ambientes de IoT são imprevisíveis , garantir a disponibilidade dos dados e a tolerância a falhas é crucial para um gerenciamento eficiente.
- **A detecção e prevenção de fraudes**, bem como **a gestão de riscos**, são tarefas cruciais para empresas de todos os setores. O Cassandra desempenha um papel vital nessas áreas, permitindo a análise em tempo real de grandes conjuntos de dados.
- **Sistemas de gerenciamento de conteúdo e plataformas de publicação digital** exigem bancos de dados robustos e escaláveis para lidar com altos volumes de gravação, garantir a disponibilidade dos dados e manter a consistência. O Cassandra oferece todos esses recursos essenciais.
- Setores que dependem da **análise de dados coletados em sequência** podem se beneficiar dos recursos do Cassandra para lidar com grandes quantidades de informações e realizar consultas complexas para descobrir tendências, padrões e insights. Com sua arquitetura distribuída, a modelagem de dados de séries temporais do Cassandra é capaz de capturar, armazenar e consultar com eficiência esse tipo de dados sequenciais.

Histórico de Desenvolvimento

2007

Remonta a um projeto do Facebook em 2007 para impulsionar a pesquisa na caixa de entrada, **inspirado nos bancos de dados Dynamo da Amazon e Bigtable do Google**, desenvolvido por Avinash Lakshman, couator do Dynamo da Amazon, e Prashant Malik

2008

Em em julho de 2008, o Facebook tornou o projeto open source

2009

Em março de 2009, o Cassandra foi adotado pela **Apache Foundation**, tornando-se um projeto de alto nível (Top-Level Project)

2010

Em fevereiro de 2010, foi oficialmente promovido para um **projeto de nível superior**, um marco importante para a governança e adoção pela comunidade

PRESENTE

Desde então, evoluiu rapidamente e se tornou **um dos principais bancos NoSQL distribuídos do mundo**

Utilização por Empresas

Muitas grandes empresas usam o Cassandra devido à sua escalabilidade e alta disponibilidade. Alguns exemplos notáveis:

- **Netflix:** Armazena histórico de visualizações, recomendações e registros de streaming em tempo real
- **Instagram:** Armazena dados de curtidas, seguidores e atividades de usuários
- **Spotify:** Utiliza para gerenciamento de metadados e playlists
- **Uber:** Para rastreamento em tempo real e armazenamento de dados de viagens
- **Ebay:** Para logs e histórico de transações
- **Apple:** Tem mais de 75.000 instâncias de Cassandra para iCloud e iMessage

Quando é vantajoso utiliza-lo

É largamente utilizado em aplicações de missão crítica onde a segurança e disponibilidade dos dados são fatores críticos junto com baixíssima latência para inserção e acesso de dados.

- Quando o sistema precisa estar disponível 24 horas por dia, sem ponto único de falha.
- Quando há grande volume de dados distribuídos em diferentes regiões.
- Quando o crescimento de dados é imprevisível, exigindo escalabilidade fácil e rápida.
- Quando o foco é desempenho em leitura e gravação distribuída, e não em transações complexas.

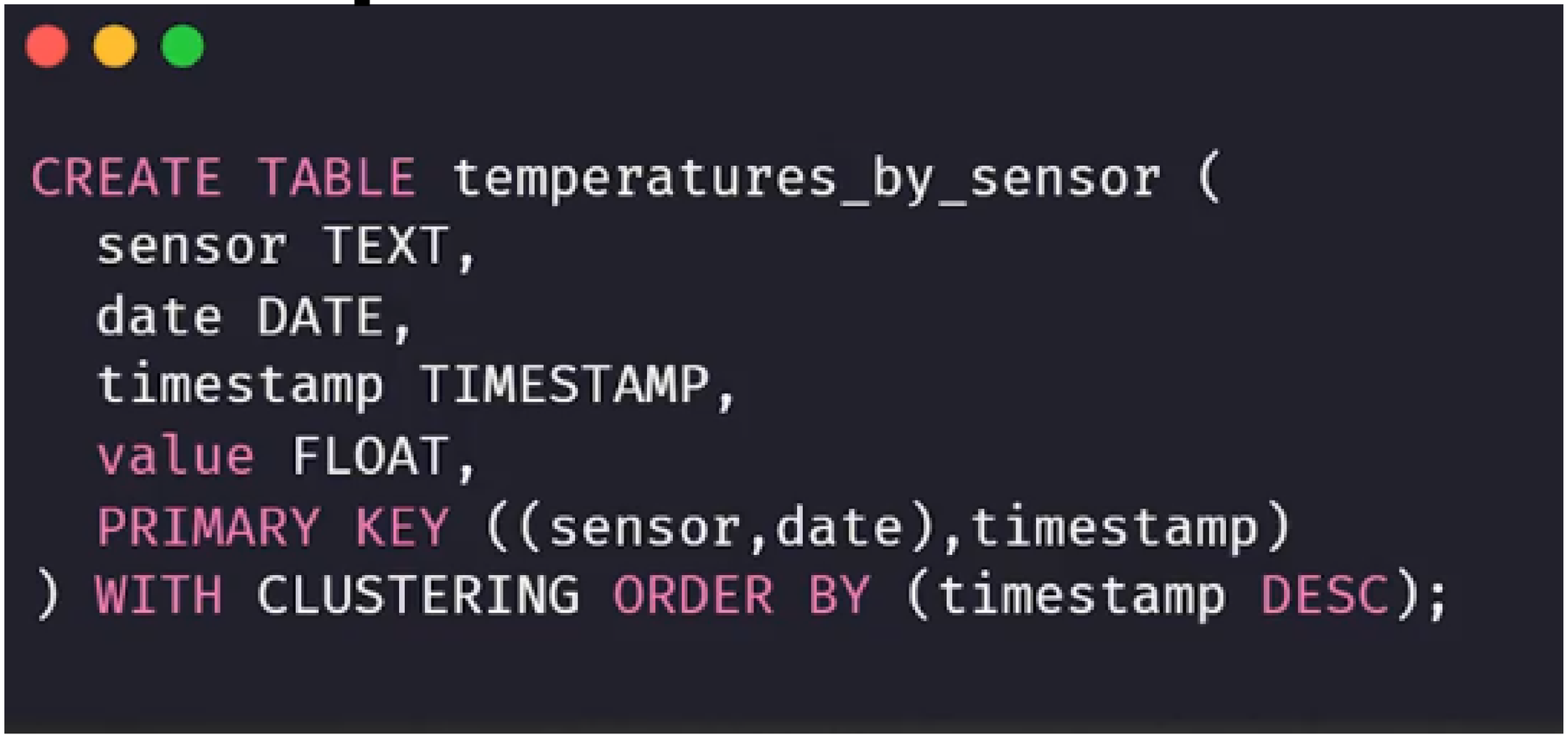
Situações em que o Uso é Desaconselhável

Evite o Cassandra quando:

- Você precisa de transações complexas (ACID) — como as de bancos, contabilidade ou ERP
- Há muitas relações entre tabelas (JOINs e integridade referencial)
- É preciso fazer consultas analíticas complexas ou relatórios com agregações pesadas
- A equipe não tem familiaridade com modelos NoSQL — a curva de aprendizado é diferente

Breve tutorial de uso

exemplo de aplicação que armazena temperaturas de sensores



```
CREATE TABLE temperatures_by_sensor (
   sensor TEXT,
   date DATE,
   timestamp TIMESTAMP,
   value FLOAT,
   PRIMARY KEY ((sensor,date),timestamp)
) WITH CLUSTERING ORDER BY (timestamp DESC);
```

Breve tutorial de uso

```sql
SELECT sensor, date, timestamp, value
FROM temperatures_by_sensor
WHERE sensor = 'S001'
  AND date = '2023-01-01'
LIMIT 5
```

Breve tutorial de uso

```
CREATE KEYSPACE my_keyspace
  WITH replication = {'class': 'SimpleStrategy', 'replication_factor': 1};

USE my_keyspace;

CREATE TABLE users (
    user_id uuid,
    name text,
    email text,
    age int,
    PRIMARY KEY (user_id)
);
```

Breve tutorial de uso

```
INSERT INTO
  users (user_id, name, email, age)
VALUES
  (uuid(), 'Vanessa', 'vanessa@gmail.com', 25);
```

OBRIGADO!