

# Capítulo 3 – HBase

---

## Introdução

O **HBase** é um banco de dados NoSQL orientado a colunas, inspirado no Bigtable do Google. Ideal para **Big Data**, armazena bilhões de linhas em clusters distribuídos.

- Dados organizados em **row keys** (chaves de linha).
  - Agrupados em **famílias de colunas**.
  - Cada célula pode ter múltiplas versões (timestamps).
  - Tabelas são **esparsas**: sem necessidade de colunas nulas.
- 

## Dia 1 – CRUD e Administração de Tabelas

### Criando tabela

```
create 'usuarios', 'perfil', 'atividade'
```

### Inserindo dados

```
put 'usuarios', 'user1', 'perfil:nome', 'João Silva'
put 'usuarios', 'user1', 'perfil:idade', '29'
put 'usuarios', 'user1', 'atividade:posts', '134'
```

### Consultando

```
get 'usuarios', 'user1'
get 'usuarios', 'user1', 'perfil:nome'
scan 'usuarios'
```

### Atualizando (nova versão)

```
put 'usuarios', 'user1', 'perfil:idade', '30'
```

### Deletando

```
delete 'usuarios', 'user1', 'perfil:idade'
deleteall 'usuarios', 'user1'
```

---

## Dia 2 – Trabalhando com Big Data

- **Regiões e RegionServers** → tabelas divididas automaticamente.
- **Row keys bem projetadas** evitam hotspots.
- **Bulk load** com HFiles é mais eficiente que inserir linha a linha.
- **Filtros** para buscas em scans.

Exemplo filtro:

```
scan 'usuarios', { FILTER => "ValueFilter(=, 'binary:João Silva')" }
```

---

## Dia 3 – Levando para a Nuvem

### Arquitetura

- **HMaster**: coordena.
- **RegionServers**: armazenam regiões.
- **ZooKeeper**: gerencia disponibilidade.

### Tolerância a falhas

- Dados armazenados no HDFS (com replicação).
- Se um RegionServer cai, o HMaster redistribui.

### HBase na nuvem

- Amazon EMR, Google Bigtable, Cloudera, Hortonworks.

### Segurança

- Autenticação via Kerberos.
- ACLs e auditoria de logs.

---

## Wrap-Up

- HBase = ótimo para dados massivos distribuídos.
- Não possui SQL ou JOINS.
- Exige **planejamento de row keys**.
- Perfeito para logs, sensores e eventos em escala.