UFSC-CTC-INE-PPGCC

INE 410131 – Gerencia de Dados para Big Data

Aula 3 – Introdução a Bancos de Dados NoSQL

Computação na Nuvem

- Paradigma de oferta de serviços remotos de computação
  - serviços oferecidos geralmente através da *Internet*

Computação na Nuvem

- Objetivos
  - Atendimento em larga escala de usuários e organizações sem infraestrutura computacional ou capital
  - Atendimento de organizações com requisitos dinâmicos em termos de demanda computacional
- Características principais dos serviços oferecidos
  - Baixo custo (ou mesmo gratuitos)
  - Transparência de acesso
  - Elasticidade (extensão/retração de serviços sob demanda)
- Analogia com serviços de luz, água, telefone, ...

Níveis de Oferta de Serviços

- Hardware (Infraestrutura) (IaaS)
  - Servidores, disco, rede, …
- Plataformas (PaaS)
  - SO, ambientes de desenvolvimento, linguagens de programação
- Software (SaaS)
  - Propósitos específicos e execução em diferentes dispositivos (*notebooks*, celulares,...)
- Gerência de Dados (DBaaS)
  - **SGBDs nas nuvens**
    - Exemplos: *Amazon Dynamo, Apache Cassandra, Mongo DB, VoltDB, ...*

BDs nas Nuvens - Características

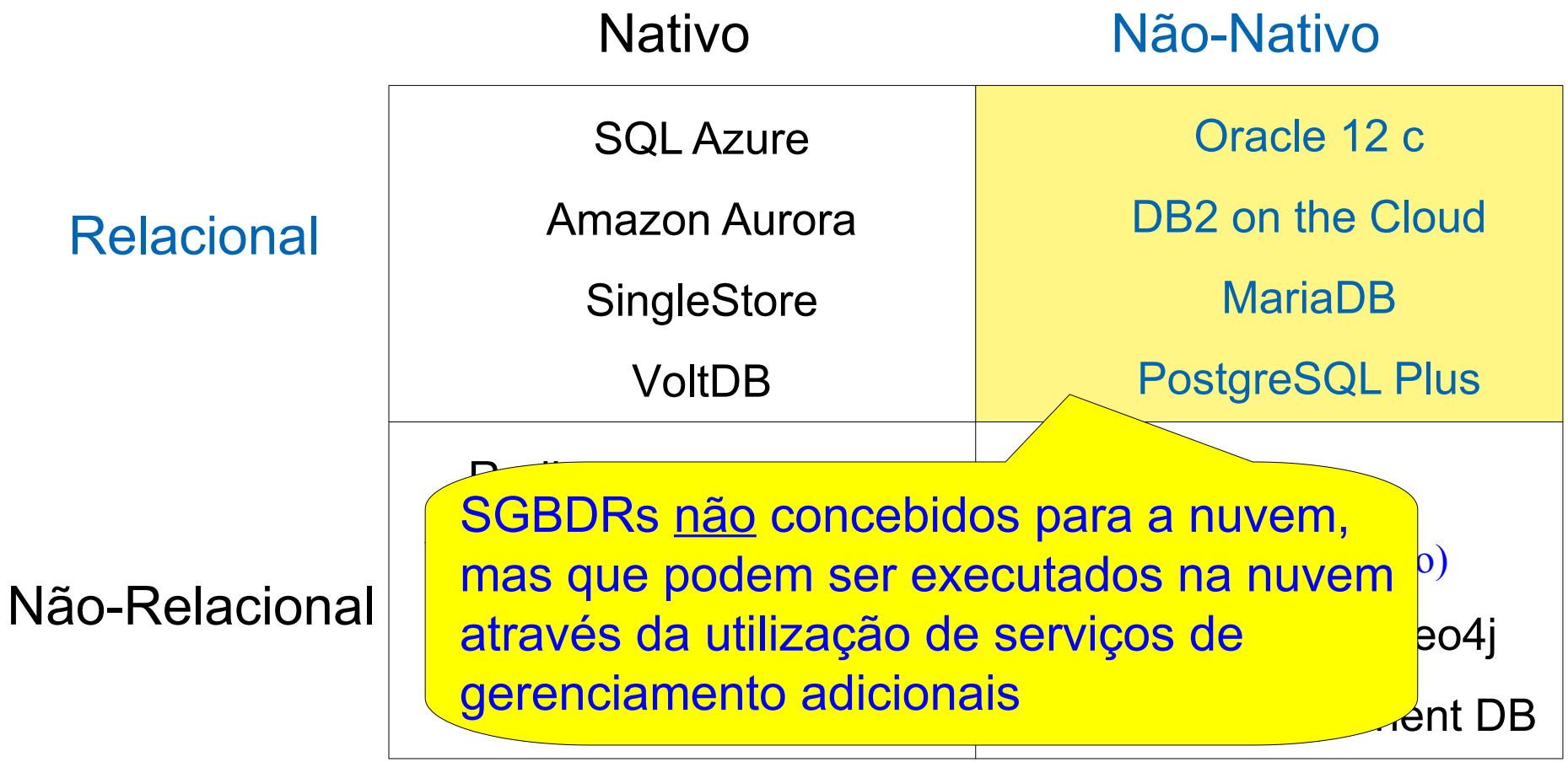
- Escalabilidade
  - Suporte à *Big Data (volume, velocidade, variedade, ...)*
  - HW robusto/ SW de processamento de alto desempenho
  - Elasticidade na demanda por operações sobre dados e alocação de recursos (escalabilidade horizontal)
- Disponibilidade
  - Replicação de dados, consistência relaxada (ACID x BASE)
- APIs simples para acesso
  - Baixo *overhead* com *parsing*/otimização/execução de comandos de linguagens de BD
- Delegação de tarefas de administração de dados
  - Exemplos: *tuning, backup, recovery, ...* do BD
- Redução de custos com aquisição, treinamento e manutenção de um SGBD local

Categorias de BD na Nuvem

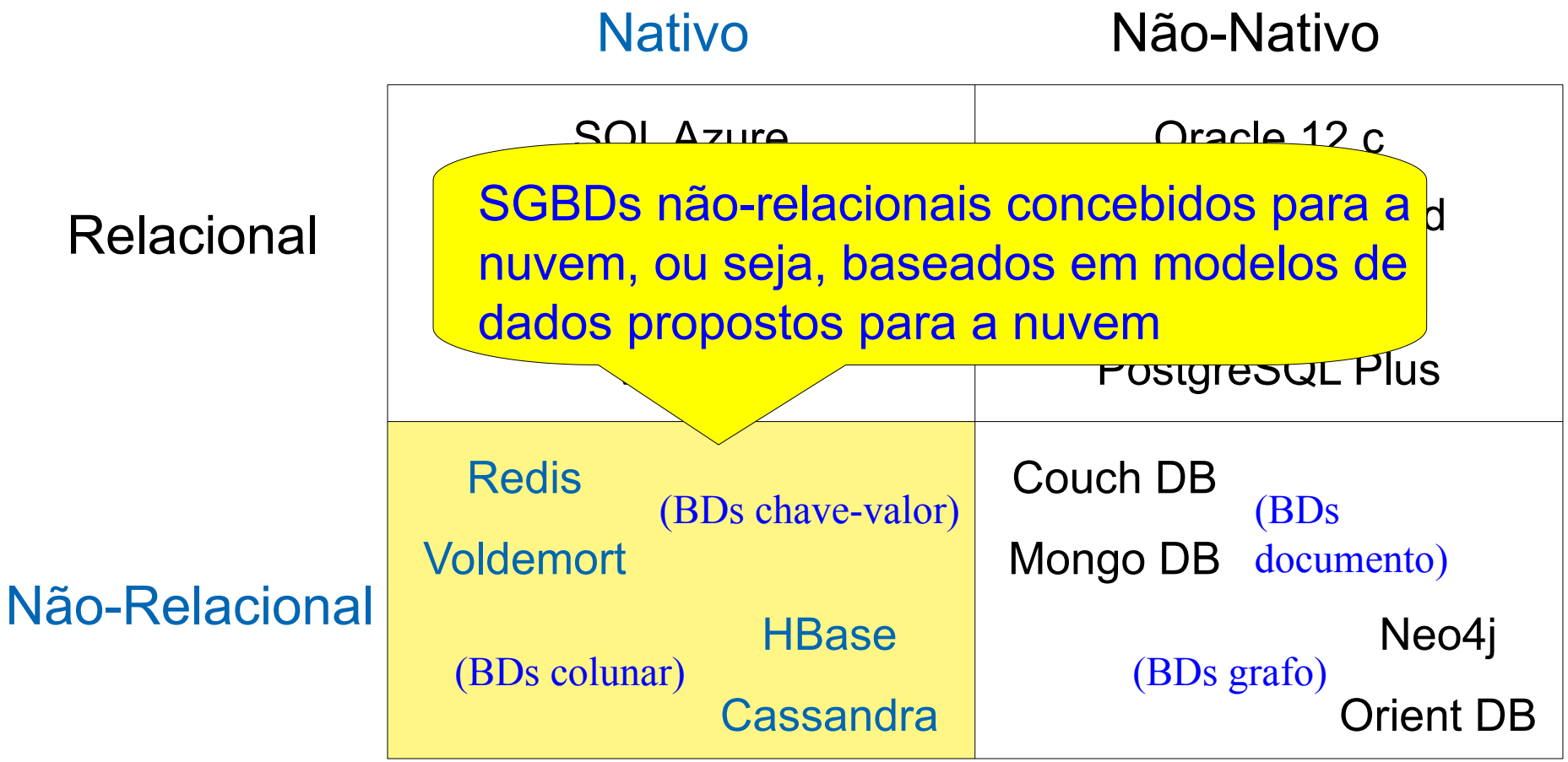
	Nativo	Não-Nativo
Relacional	SQL Azure Amazon Aurora SingleStore VoltDB	Oracle 12c DB2 on the Cloud MariaDB PostgreSQL Plus
Não-Relacional	Redis Voldemort (BD chave-valor) HBase Cassandra (BD colunar)	Couch DB Mongo DB (BD documento) Neo4j Orient DB (BD grafo)

Categorias de BD na Nuvem

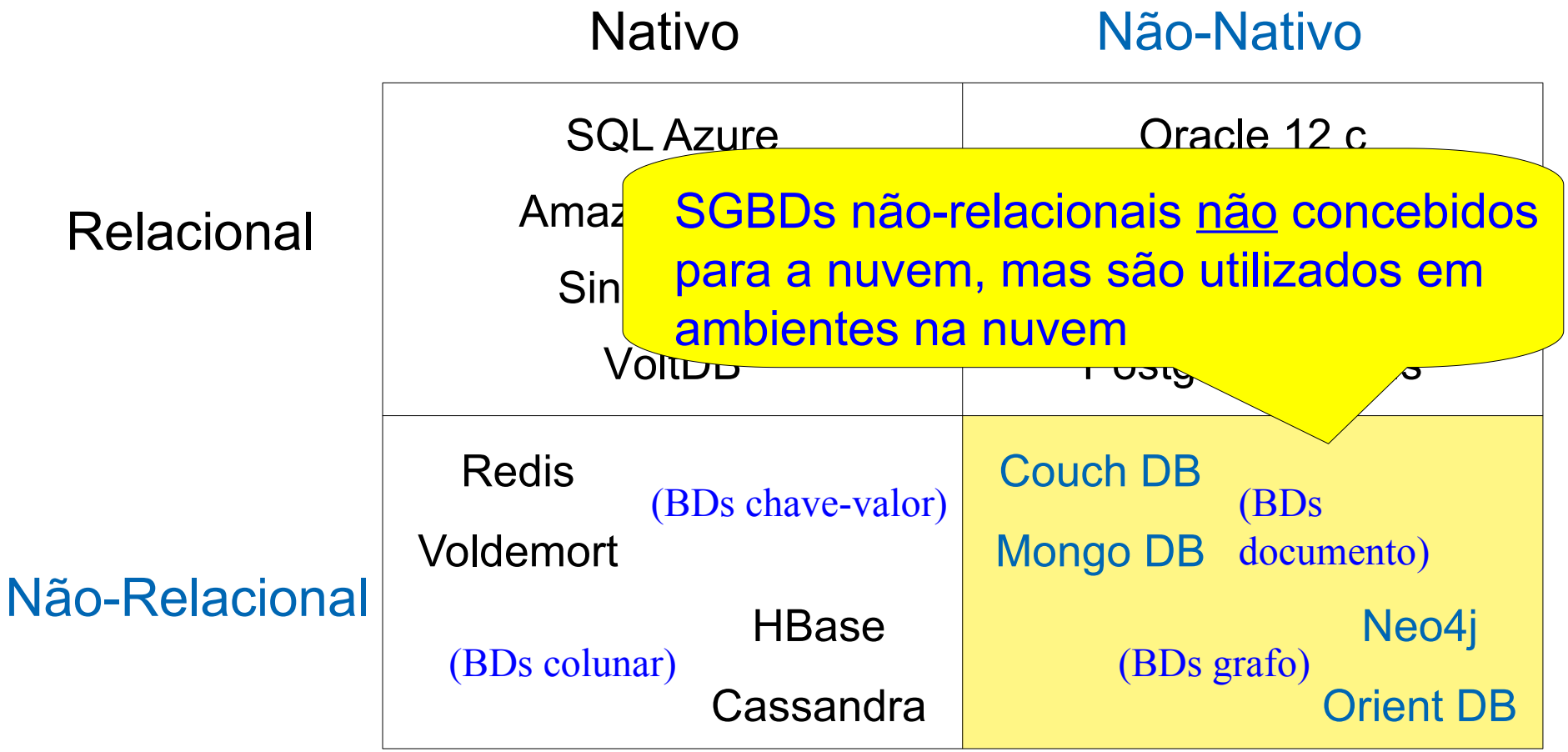
Categorias de BD na Nuvem



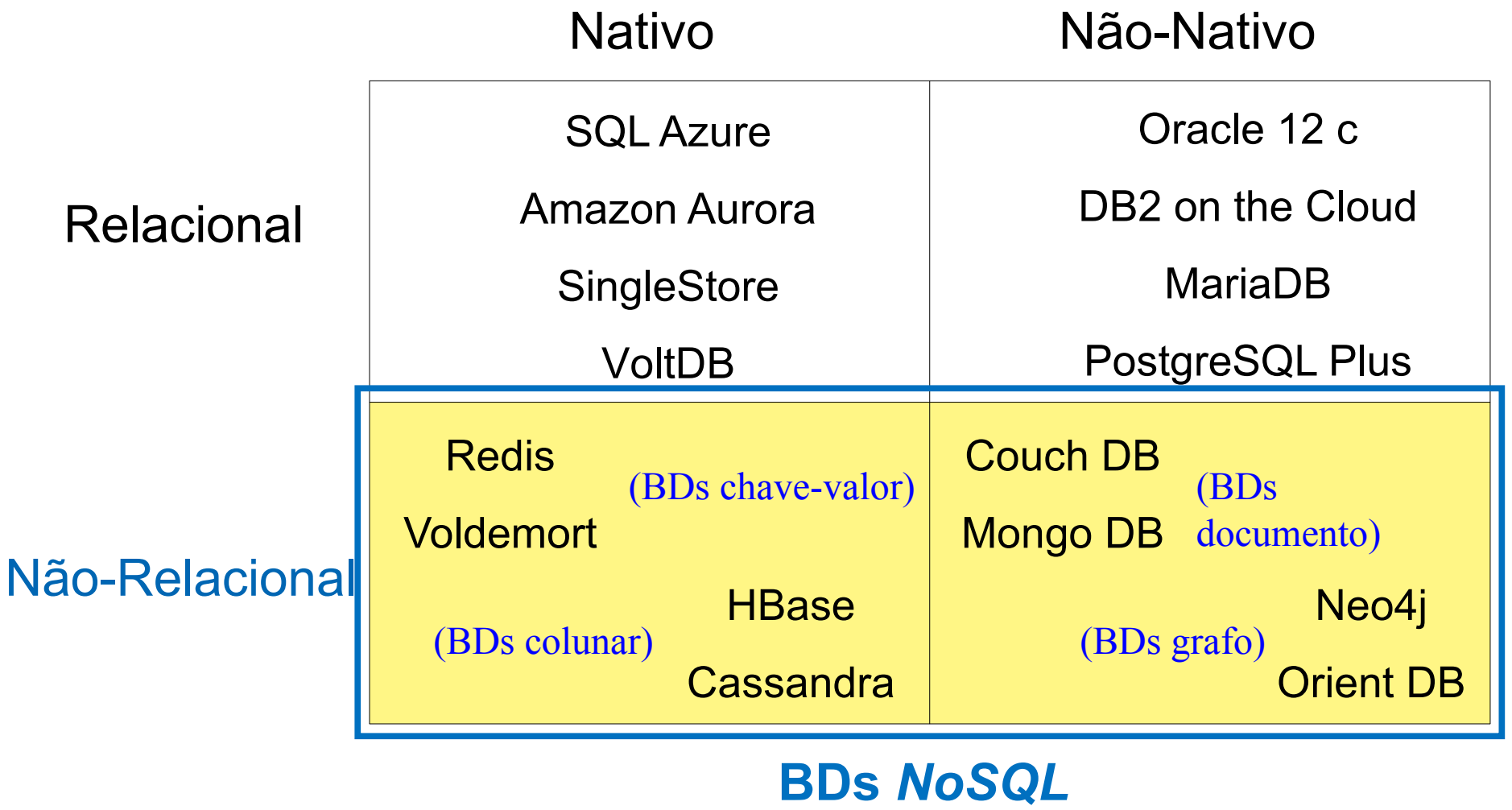
Categorias de BD na Nuvem



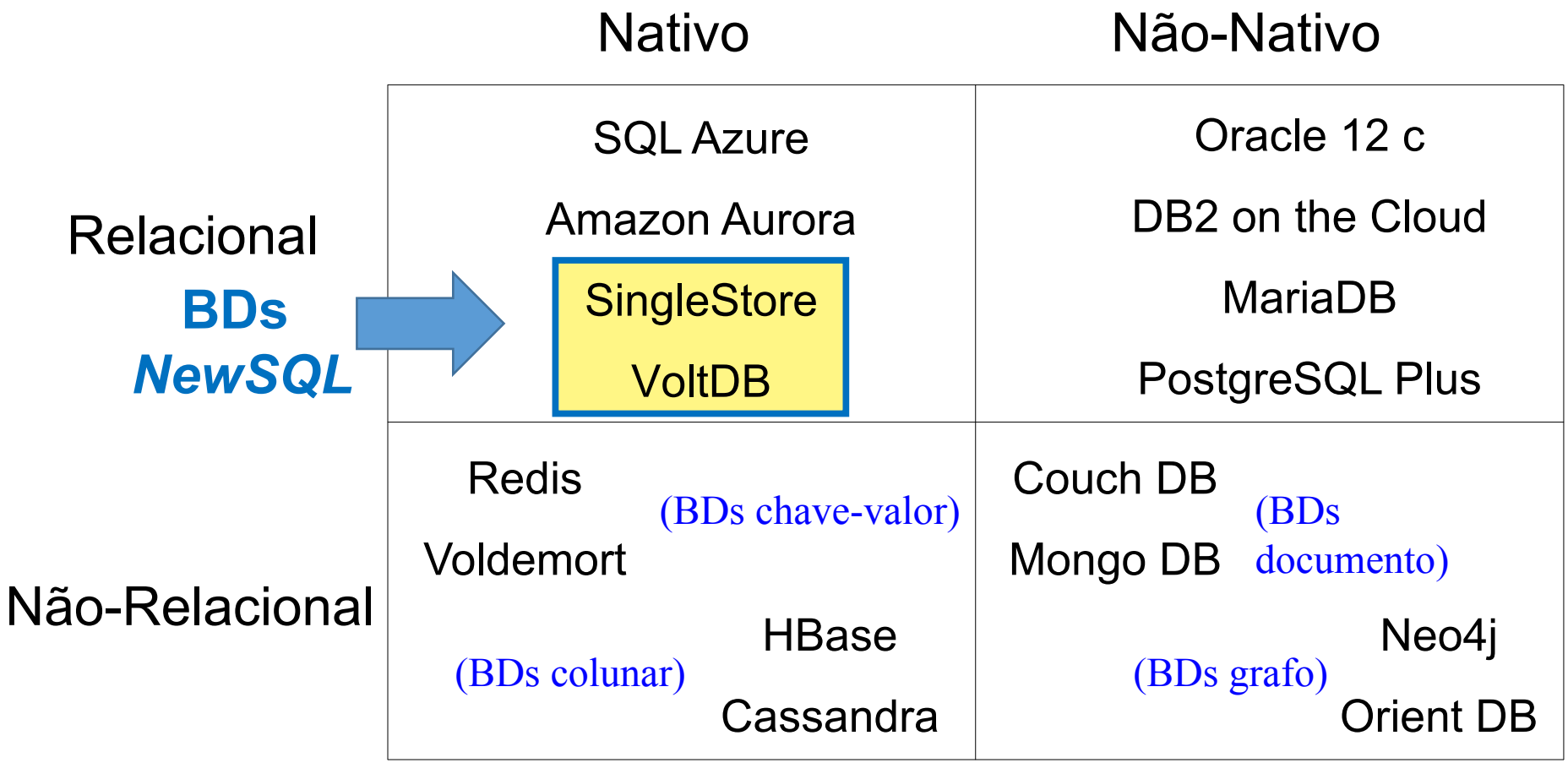
Categorias de BD na Nuvem



Categorias de BD na Nuvem



Categorias de BD na Nuvem

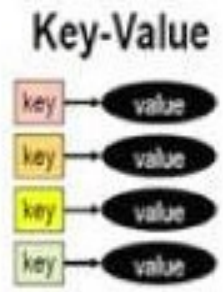


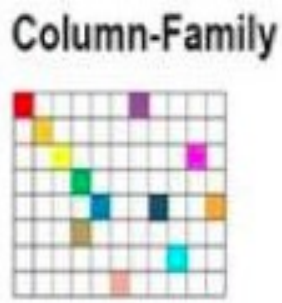
NoSQL (Not Only SQL)

- Movimento pelo desenvolvimento de SGBDs não-relacionais para o gerenciamento de dados na nuvem
- Principais características
  - Métodos de acesso simples
    - APIs baseadas em acesso por chave
  - Não suporta junções
  - Maioria não suporta relacionamentos entre dados e integridade referencial
  - Modelos de dados heterogêneos
    - Ausência de esquema (*schemaless*)
    - Esquema opcional
    - Esquemas flexíveis

BDs NoSQL

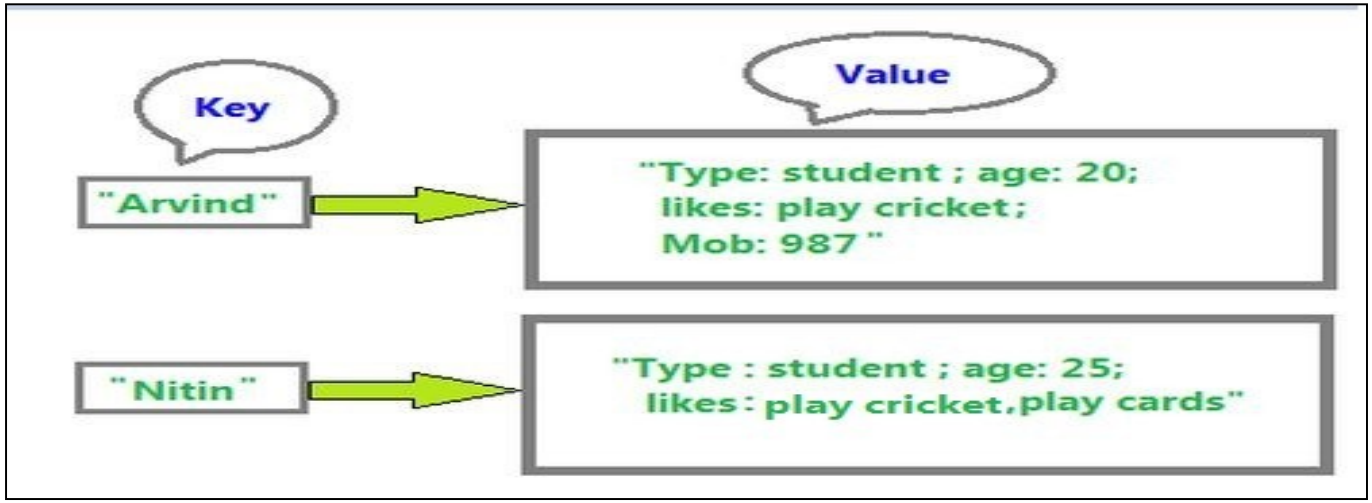
- Quatro modelos de dados
  - BD chave-valor
    - Exemplos: Redis, Voldemort
  - BD colunar
    - Exemplos: Cassandra, HBase
  - BD de documento
    - Exemplos: CouchDB, MongoDB
  - BD de grafo
    - Exemplos: Neo4j, Orient DB





BD Chave-Valor

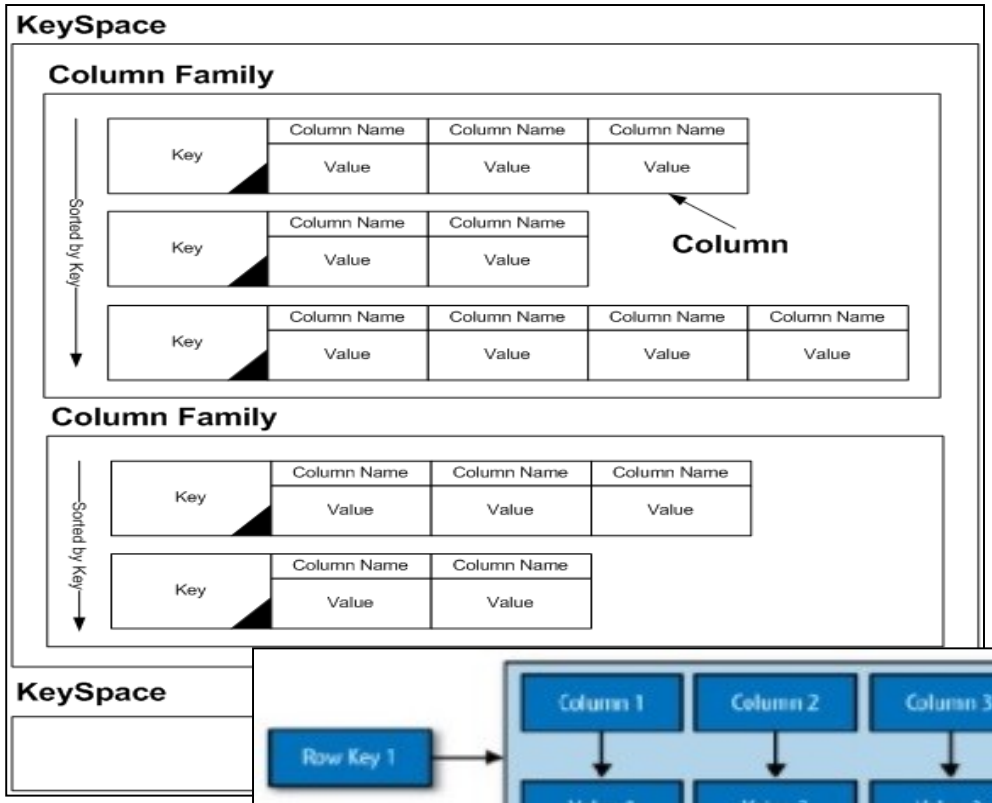
- Modelo simples similar a uma estrutura de indexação
  - Chave identifica um conteúdo mono ou multivalorado
- API simples
  - *get(key)*, *put(key, value)*, *delete(key)*
- Não suporta
  - Definição de esquemas
  - Relacionamentos entre dados
  - Linguagem de consulta

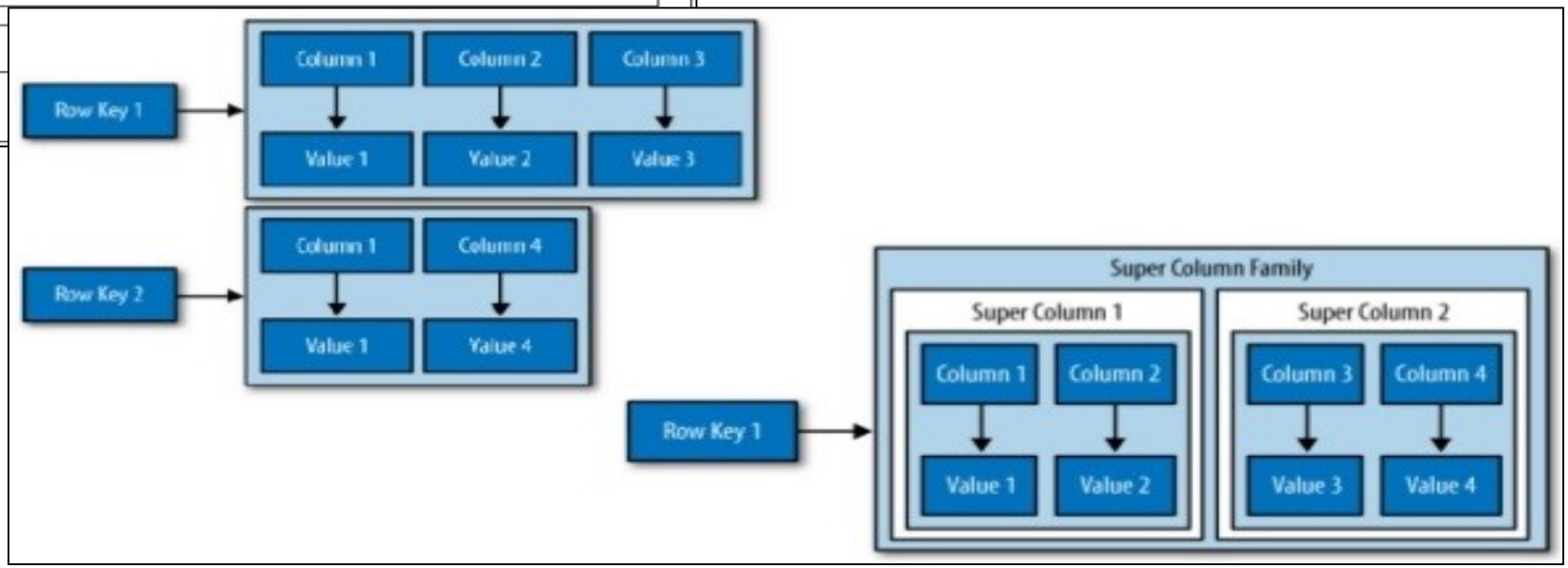


BD Colunar

- Modelo mais complexo que o chave-valor
  - Conceitos: *keyspace* (≡ BD), *column family* (≡ tabela) e um *conjunto de colunas* (≡ registro)
  - Uma coluna possui um *nome* e um *valor*
  - Um conjunto de colunas é acessado por uma *chave*
  - Itens de dados (“registros”) podem ter colunas diferentes
  - Suporte a colunas multivaloradas e supercolunas
- APIs proprietárias e/ou linguagens de consulta simples
- Não suporta relacionamentos entre dados

BD Colunar


KeySpace
Column Family
Key
Column Name
Value
Column
Sorted by Key
Column Family
KeySpace


Row Key 1
Column 1
Column 2
Column 3
Value 1
Value 2
Value 3
Row Key 2
Column 1
Column 4
Value 1
Value 4
Row Key 1
Super Column Family
Super Column 1
Super Column 2
Column 1
Column 2
Column 3
Column 4
Value 1
Value 2
Value 3
Value 4

BD Colunar - Exemplos

- Hbase API

```
... /* Inclusão de uma coluna em uma família de colunas */
HColumnDescriptor coluna = new HColumnDescriptor("CPF");
Pessoa.modifyFamily(coluna); ...
```

- Cassandra CQL

```
CREATE KEYSPACE RedeSocial
WITH replication = {'class': 'SimpleStrategy',
                    'replication_factor' = 3};
USE RedeSocial;

CREATE TABLE Pessoa (  /* similar a uma família de colunas */
ID int, nome text STATIC, sexo varchar, DN date PRIMARY KEY(ID));
```

coluna obrigatória em todas as tuplas

```
DELETE sexo FROM Pessoa /* remoção de uma coluna de uma tupla */
WHERE ID = 335;
```

BD de Documentos

- Modelo adequado à representação de objetos complexos
  - Um objeto (“documento”) possui uma chave e um conjunto de atributos
  - Atributos podem ter domínios atômicos ou complexos (listas, tuplas, conjuntos)
  - Conjuntos de documentos são mantidos em coleções
- APIs proprietárias e/ou linguagens de consulta simples
- Não suporta relacionamentos entre dados
  - Falta de padronização

Exemplos: MongoDB (JSON), Amazon SimpleDB (Domínio→Item →Atributo →{Valor})

BD CouchDB – Documento JSON


```
{ "_id":"discussion_tables",
 "_rev":"D1C946B7",
 "Sunrise":true,
 "Sunset":false,
 "FullHours":[1,2,3,4,5,6,7,8,9,10],
 "Activities": [
   {"Name":"Football", "Duration":2, "DurationUnit":"Hours"},
   {"Name":"Breakfast", "Duration":40, "DurationUnit":"Minutes",
         "Attendees":["Jan", "Damien", "Laura", "Gwendolyn", "Roseanna"]} ]     }
```


Atributos *default*


```
{
 "_id":"some_doc_id",
 "_rev":"D1C946B7",
 "Subject":"I like Plankton",
 "Author":"Rusty",
 "PostedDate":"2006-08-15T17:30:12-04:00",
 "Tags":["plankton", "baseball", "decisions"],
 "Body":"I decided today that I don't like baseball. I like plankton."                    }
```

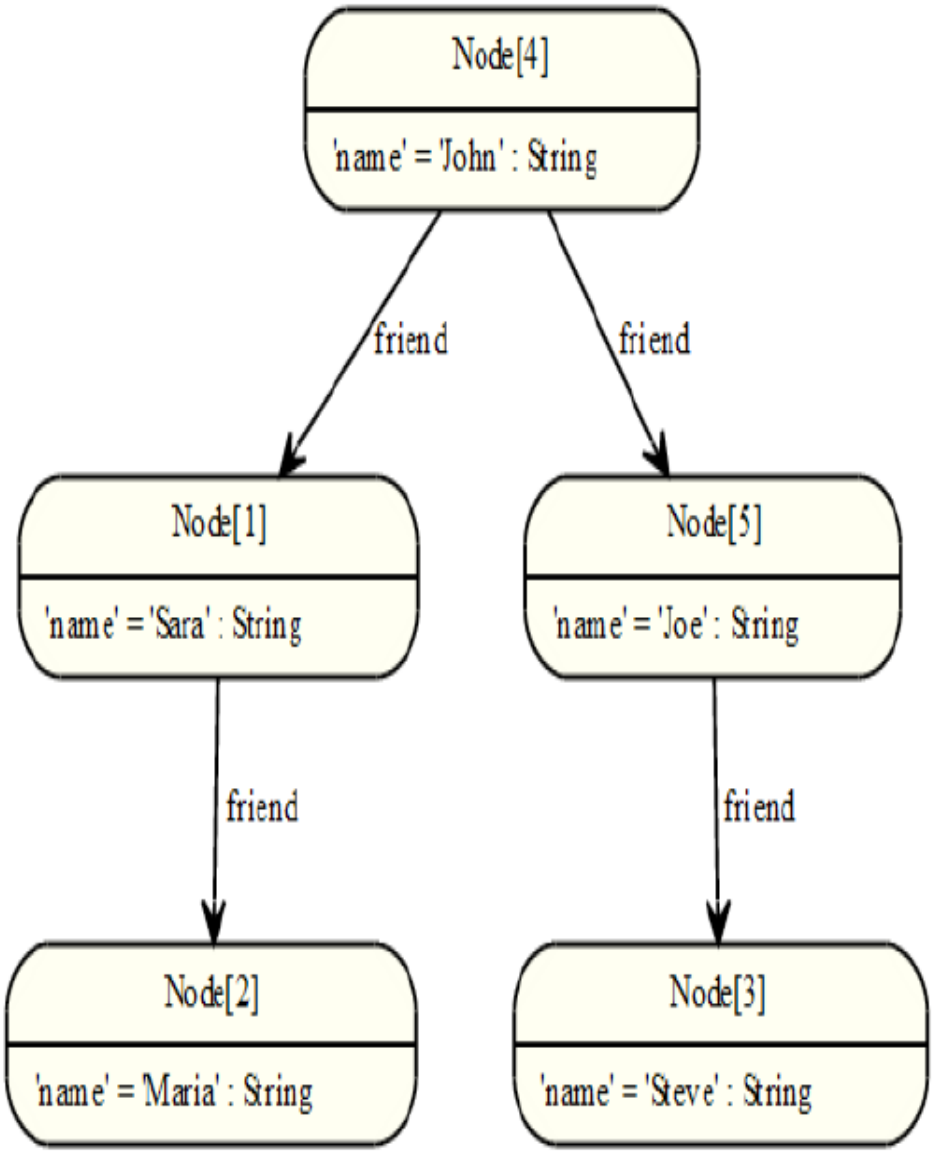
BD MongoDB – API

```
/* inserção de um documento na coleção pessoas */
/* coleção é criada, caso não exista */
db.pessoas.insert (
{ ID: 781, nome: "João da Silva", sexo: "M", idade: 50})


db.pessoas.find ({ "idade": {$gt: 40}} {ID: 1, nome: 1})


db.pessoas.replaceOne ({ ID: "781"},
{ ID: 781, nome: "Ana Souza", sexo: "F", idade: 35}))


db.cursos.updateMany ({ "depto": "INE", "centro": "CTC"},
{ $set: {"vagas": 100}))
```

BD de Grafo

- Modelo composto por nodos, arestas e atributos
  - Nodo: um item de dado (“registro”) composto por atributos
  - Aresta: relacionamento entre nodos, composto por um rótulo e atributos opcionais
  - Atributo: composto por nome e valor (domínio atômico ou multivalorado)
- APIs proprietárias e/ou linguagens de consulta

BD Neo4j – Linguagem de Consulta Cypher




```
START john=node:node_auto_index(name = 'John')
MATCH john-[:friend]->()-[:friend]->fof
RETURN john, fof
```

Resultado:

john	fof
Node[4]{name->"John"}	Node[2]{name->"Maria"}
Node[4]{name->"John"}	Node[3]{name->"Steve"}
2 rows, 2 ms	

Atividade 1 – Persistência Poliglota

- Ler o capítulo 13 do livro *NoSQL Distilled*
- Ler o texto:
  https://martinfowler.com/articles/nosql-intro-original.pdf

- Responder as seguintes perguntas:
  1) O que é persistência poliglota?
  2) Qual(is) a(s) vantagem(ns) de utilizar persistência poliglota?
  3) Qual(is) a(s) dificuldade(s) em utilizar persistência poliglota?