

UFSC-CTC-INE

INE 5600 – Bancos de Dados III

Introdução a Bancos de
Dados nas Nuvens e Bancos
de Dados NoSQL

Computação na Nuvem

- Paradigma de oferta de serviços remotos de computação
 - serviços oferecidos, em geral, 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

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

Categorias de BD na Nuvem

	Nativo	Não-Nativo
Relacional	SQL Azure Amazon Aurora MemSQL VoltDB	Oracle 12 c DB2 on the Cloud MariaDB PostgreSQL Plus
Não-Relacional	Re (have valor)	Couch DB (BDs eo4j nt DB

SGBDRs construídos para operar na nuvem, ou seja, oferecem funcionalidades de gerência de dados adequadas a um SGBD na nuvem

Categorias de BD na Nuvem

	Nativo	Não-Nativo
Relacional	SQL Azure Amazon Aurora MemSQL VoltDB	Oracle 12 c DB2 on the Cloud MariaDB PostgreSQL Plus
Não-Relacional	SGBDRs <u>não</u> concebidos para a nuvem, mas que podem ser executados na nuvem através da utilização de serviços de gerenciamento adicionais	

Categorias de BD na Nuvem

	Nativo	Não-Nativo
Relacional	SQL Azure	Oracle 12c PostgreSQL Plus
Não-Relacional	Redis (BDs chave-valor) Voldemort (BDs colunar) HBase Cassandra	Couch DB (BDs documento) Mongo DB (BDs grafo) Neo4j Orient DB

SGBDs não-relacionais concebidos para a nuvem, ou seja, baseados em modelos de dados propostos para a nuvem

Categorias de BD na Nuvem

	Nativo	Não-Nativo
Relacional	SQL Azure Amazon RDS Microsoft SQL Server VoltDB	Oracle 12c PostgreSQL
Não-Relacional	Redis (BDs chave-valor) Voldemort HBase (BDs colunar) Cassandra	Couch DB (BDs documento) Mongo DB Neo4j (BDs grafo) Orient DB

SGBDs não-relacionais não concebidos para a nuvem, mas são utilizados em ambientes na nuvem

Categorias de BD na Nuvem

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

BDs NoSQL

Categorias de BD na Nuvem

	Nativo	Não-Nativo
Relacional BDs <i>NewSQL</i>	SQL Azure Amazon Aurora MemSQL VoltDB	Oracle 12 c DB2 on the Cloud MariaDB PostgreSQL Plus
Não-Relacional	Redis Voldemort HBase Cassandra (BDs chave-valor) (BDs colunar)	Couch DB Mongo DB Neo4j Orient DB (BDs documento) (BDs grafo)

NoSQL (*Not Only SQL*)

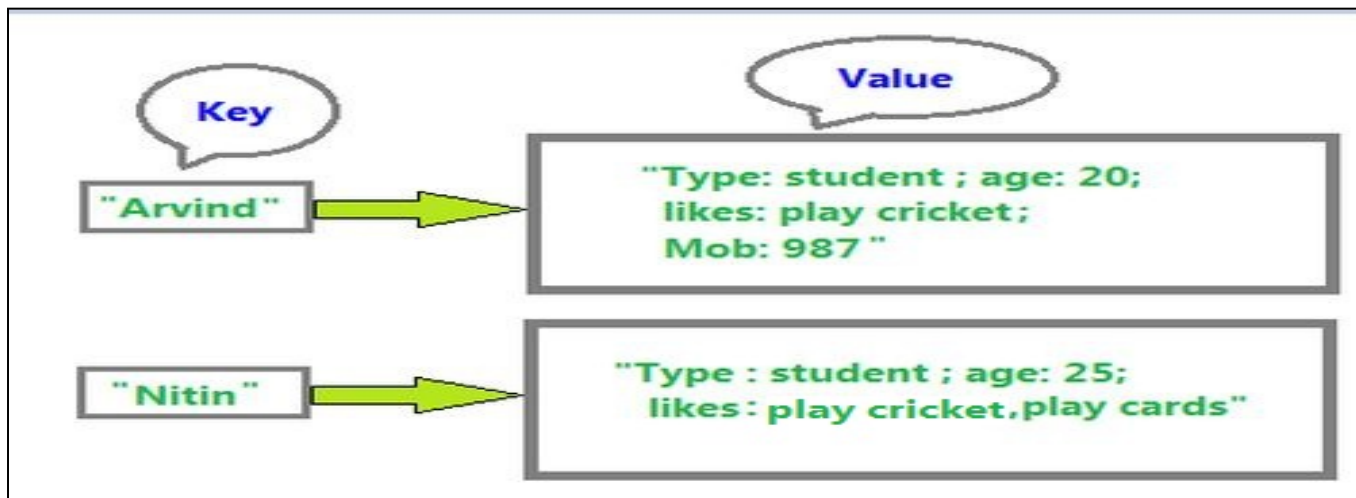
- Movimento pelo desenvolvimento de SGBDs não-relacionais para o gerenciamento de dados na nuvem
- Principais características
 - Métodos de acesso simples
 - APIs baseadas em acesso por chave
 - Não suporta junções
 - Maioria não suporta relacionamentos entre dados e integridade referencial
 - Modelos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

BDs NoSQL

- Principais famílias
 - BD chave-valor
 - Exemplos: Amazon S3, Voldemort, Redis
 - BD colunar
 - Exemplos: Cassandra, HBase
 - BD de documentos
 - Exemplos: CouchDB, MongoDB
 - BD de grafo
 - Exemplos: Neo4j, Orient DB

BD Chave-Valor

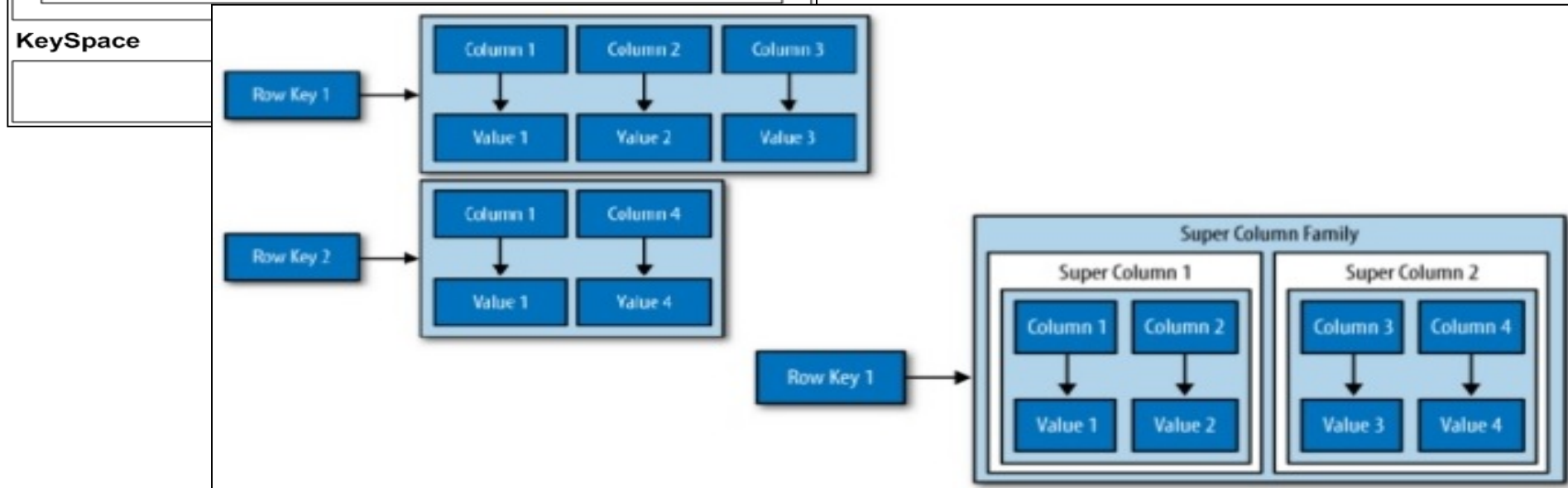
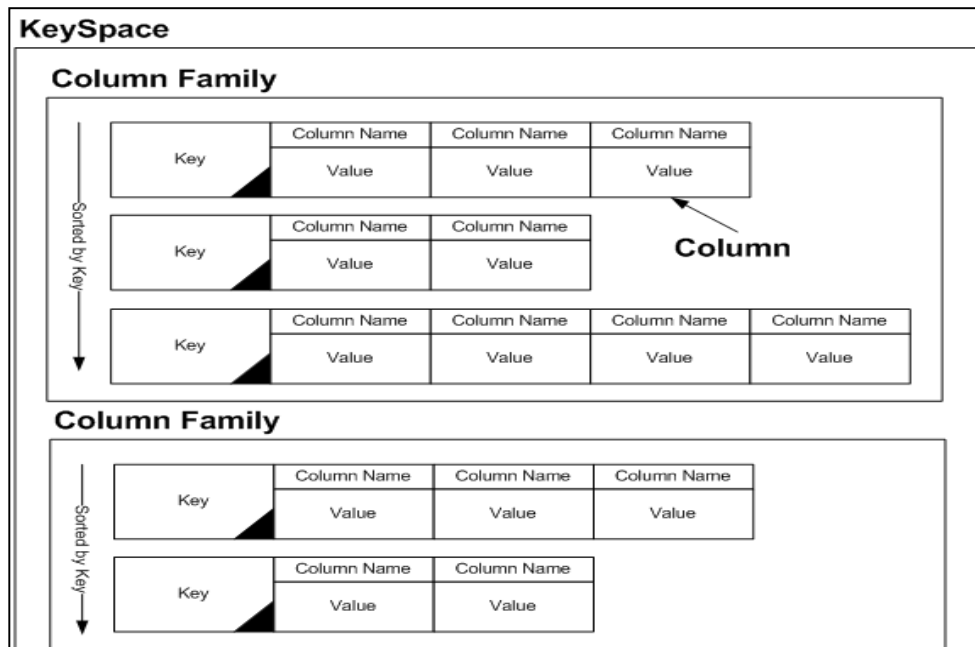
- Modelo simples similar a uma estrutura de indexação
 - Chave identifica um conteúdo mono ou multivalorado
- API simples
 - *get(key)*, *put(key, value)*, *delete(key)*
- Não suporta
 - Definição de esquemas
 - Relacionamentos entre dados
 - Linguagem de consulta



BD Colunar

- Modelo mais complexo que o chave-valor
 - Conceitos: *keyspace* (\equiv BD), *column family* (\equiv tabela) e um *conjunto de colunas* (\equiv 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



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"]} ] }  
}
```

```
{  
  "_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."  
}
```

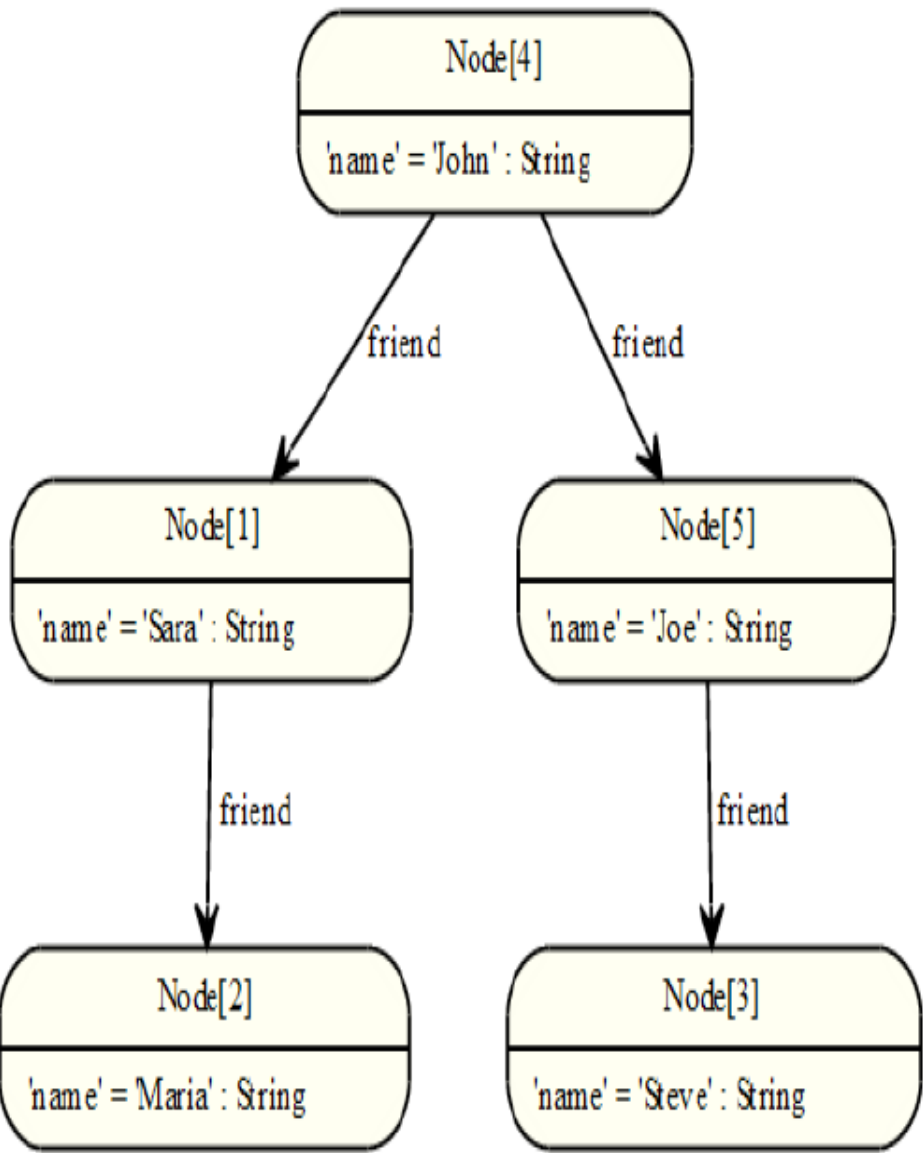
BD MongoDB – API

```
/* 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})  
  
/* busca ID e nome das pessoas com mais de 40 anos */  
db.pessoas.find ({ "idade": {$gt: 40}} {ID: 1, nome: 1})  
  
/* substitui o conteúdo de um documento */  
db.pessoas.replaceOne ({ ID: "781"},  
  { ID: 781, nome: "Ana Souza", sexo: "F", idade: 35}))  
  
/* atualização de dados de vários documentos */  
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	