

# Sistemas Distribuídos

Computação com Grandes Volumes de Dados

Prof. Tales Viegas

# Computação com Grandes Volumes de Dados

- ▶ Grandes volumes de dados pertenciam ao domínio das aplicações científicas até os anos 2000
- ▶ Com o crescimento exponencial das aplicações Web, o volume de dados que é produzido e processado tornou-se imenso
- ▶ As empresas que fornecem serviços através da Internet (como pesquisa, publicidade e redes sociais) processam diariamente gigantescas quantidade de dados

# Computação com Grandes Volumes de Dados

- ▶ Surgiu, então, o termo “Big Data”
- ▶ Coleção de dados tão grande e complexa que é difícil o seu processamento através das aplicações tradicionais
- ▶ Big Data surge em áreas como:
  - Redes de sensores, redes sociais, motores de busca
  - Serviços de vigilância (civil e militar), registros médicos
  - Arquivos de fotos e vídeo, grandes aplicações comerciais

# Computação com Grandes Volumes de Dados

- ▶ Não há muito tempo, uma aplicação ter 1000 usuários por dia era muito, e 10.000 era um caso extremo
- ▶ Hoje imensas aplicações estão hospedadas em cloud, disponíveis pela internet, suportando usuários 24h/dia
- ▶ Mais de 3,6B de pessoas hoje estão ligadas a Internet em todo o mundo (40%)
- ▶ Em 1995 era cerca de 1%
- ▶ <http://internetlivestats.com/internet-users>

# Computação com Grandes Volumes de Dados

- ▶ O número de usuários de uma aplicação é muito difícil de prever
- ▶ Uma nova app pode ir de 0 a 1M de acessos em 1 só dia
- ▶ Alguns usuários acessam uma vez e nunca mais voltam, outros acessam várias vezes ao dia

# Computação com Grandes Volumes de Dados

- ▶ Tecnologias de **cloud** suportam computação intensiva de dados através de:
  - Grande quantidade de instâncias de computação criadas on-demand
  - Sistemas de armazenamento otimizados para guardar grandes blobs (binary large objects)
  - Fornecimento de frameworks e APIs otimizadas para processar grandes quantidades de dados

# Computação com Grandes Volumes de Dados

- ▶ Alguns fornecedores de cloud foram criando tecnologias para processar grandes quantidades de dados:
  - Hadoop system
    - Hadoop Distributed File System + Map Reduce
  - Google Map Reduce
    - Google File System + Map Reduce
  - ...
    - Sistema de Armazenamento + Sistema de Computação

# Big Data e Bases de Dados

- ▶ Tradicionalmente as bases de dados relacionais distribuídas foram consideradas a evolução natural para o aumento do volume de dados

# Big Data e Bases de Dados

- ▶ Uma base de dados distribuída é constituída dividindo os dados por várias localizações
- ▶ São sistemas robustos, com suporte para transações distribuídas, mas são pouco eficientes para:
  - Enormes quantidades de dados
  - Dados não estruturados
  - Dados cuja estrutura evolui ao longo do tempo

# Big Data e Bases de Dados

- ▶ Empresas como Google, Facebook, Amazon, entre outras, concluíram que o modelo relacional de base de dados não servia para suportar as necessidades das suas aplicações
- ▶ Surgiram então as bases de dados NoSQL
  - Modelo de dados mais flexível
  - Capacidade de escalar dinamicamente para suportar um maior número de usuários
  - Pesquisas mais eficientes

# Sistemas de Arquivos Distribuídos

- ▶ A necessidade de novas formas de armazenar grandes quantidades de dados levou ao surgimento de Sistemas de Arquivos Distribuídos e de novos sistemas de Bases de Dados
  - Google FS
  - Amazon S3 (Simple Storage Service)
  - GlusterFS

# Sistemas NoSQL

- ▶ NoSQL – Not Only SQL – Termo criado em 1998 para identificar uma base de dados que não oferecia uma interface SQL para manipular e acessar os dados
- ▶ Não são bases de dados relacionais, mas uma coleção de scripts que permitem executar as operações mais comuns em bases de dados, utilizando geralmente arquivos para armazenar a informação

# Sistemas NoSQL

- ▶ A filosofia é ultrapassar as limitações impostas pelo modelo relacional e criar sistemas mais eficientes

# Sistemas NoSQL

- ▶ Para isso são usadas tabelas sem esquemas fixos, que permitem:
  - Contem um maior número de tipos de dados
  - Evitar operações de junção para aumentar o desempenho e permitir escalar o sistema horizontalmente
- ▶ Existem muitos modelos:
  - Document Stores; Graphs; Key–Value Stores
  - Multivalue Databases; Object Databases
  - Tabular Stores; Tuple Stores

# SQL vs NoSQL

- ▶ Características de uma Base de Dados SQL:
  - Dados são armazenados em tabelas
  - Associações entre os dados são representadas por dados
  - Linguagem de Manipulação de Dados (DML)
  - Linguagem de Definição de Dados (DDL)
  - Transações

# SQL vs NoSQL

## ▶ Características de uma Base de Dados SQL:

- Aplicações especificam o que querem e não “como querem”
- Existe um motor de otimização de consultas
- Abstração do nível físico: a camada física pode mudar sem ser necessário mudar as aplicações
- Criação de índices para suportar as consultas

# SQL vs NoSQL

## ▶ Características de uma Base de Dados SQL:

- Uma transação deve respeitar as propriedades ACID:
  - Atomic – todas as operações de uma operação são completadas (commit) ou nenhuma delas
  - Consistent – uma transação transforma uma base de dados de um estado consistente em outro estado consistente. Esta consistência é definida através de restrições

# SQL vs NoSQL

## ▶ Características de uma Base de Dados SQL:

- Uma transação deve respeitar as propriedades ACID:
  - Isolated – os resultados das alterações efetuadas durante uma transação não são visíveis até que a transação termine com sucesso
  - Durable – os resultados de uma transação terminada com sucesso (committed) sobrevivem a falhas

# SQL vs NoSQL

- ▶ Características de uma Base de Dados NoSQL:
  - Grandes volumes de dados
  - Replicação e distribuição escaláveis
    - Potencialmente milhares de máquinas
    - Potencialmente distribuída por todo o mundo
  - Resposta rápida a consultas
  - Principalmente consultas, poucos updates

# NoSQL - Características Distintivas

- ▶ Sem um esquema fixo
- ▶ Transações não são ACID, são BASE
- ▶ Teorema de CAP
- ▶ Open-Source

# Transações BASE

- ▶ Oposto de ACID
- ▶ Basically Available
- ▶ Soft State – o estado do sistema pode mudar ao longo do tempo, mesmo sem input
- ▶ Eventually Consistent – o sistema vai tornar-se consistente ao longo do tempo

# Características NoSQL

- ▶ Consistência fraca
- ▶ Disponibilidade em primeiro lugar
- ▶ “Best Effort”
- ▶ Respostas aproximadas
- ▶ Fácil e rápido

# Teorema de CAP (Brewer, 2000)

- ▶ Um sistema distribuído pode apenas suportar duas das seguintes características:
  - Consistência
  - Disponibilidade
  - Tolerância a partições
- ▶ Consistência
  - Todos os nós vêem os mesmos dados ao mesmo tempo
  - O cliente percebe que um conjunto de operações ocorreu como um todo
  - Equivale ao conceito de Atomicidade das transações ACID

# Teorema de CAP (Brewer, 2000)

- ▶ Disponibilidade
  - A avaria de um nó não faz com que os outros deixem de funcionar
  - Cada operação deve terminar e dar a resposta pretendida
- ▶ Tolerância a Partições
  - O sistema continua mesmo que haja mensagens perdidas
  - As operações serão concluídas mesmo que algum componente individual não esteja disponível

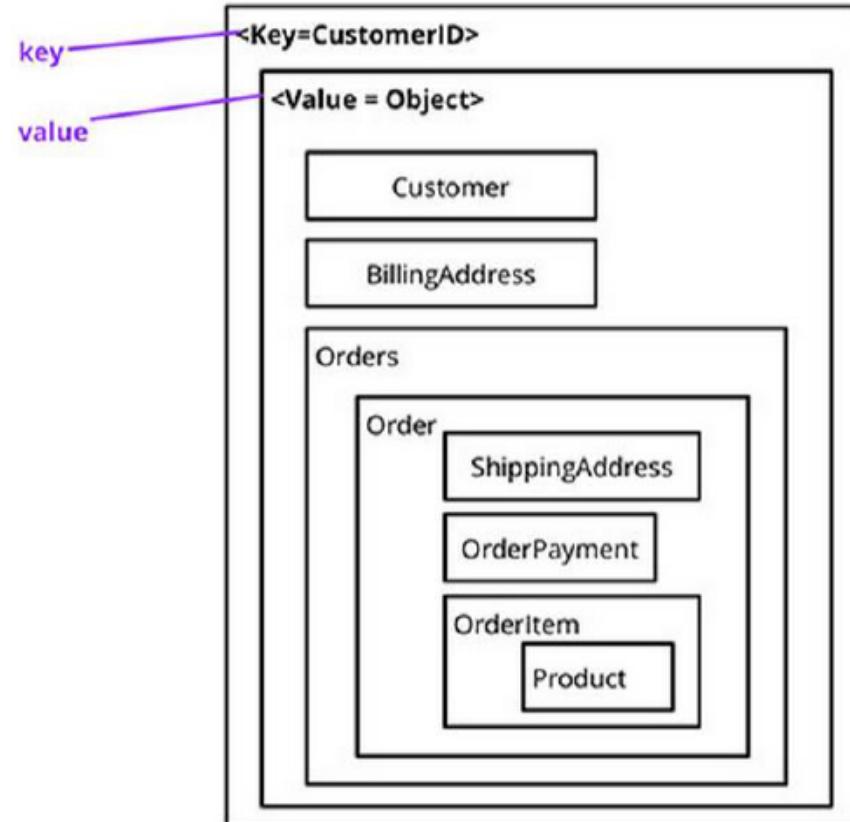
# Tipos de Bases NoSQL

## ► As mais importantes:

- Key-Value Databases
- Document Databases
- Column Family Stores
- Graph Databases

# NoSQL – Key–Value Databases

- ▶ Modelo NoSQL mais simples de usar
- ▶ O cliente pode:
  - Obter o valor para uma chave
  - Atribuir um valor para uma chave
  - Apagar uma chave da base de dados



# NoSQL – Key–Value Databases

- ▶ O valor é um Blob armazenado na base de dados
- ▶ O sistema de BD não conhece o que está dentro do Blob
- ▶ É da responsabilidade da aplicação conhecer o que foi armazenado
- ▶ Os valores são acessados pela única chave (primária). O acesso é rápido e facilmente escalável

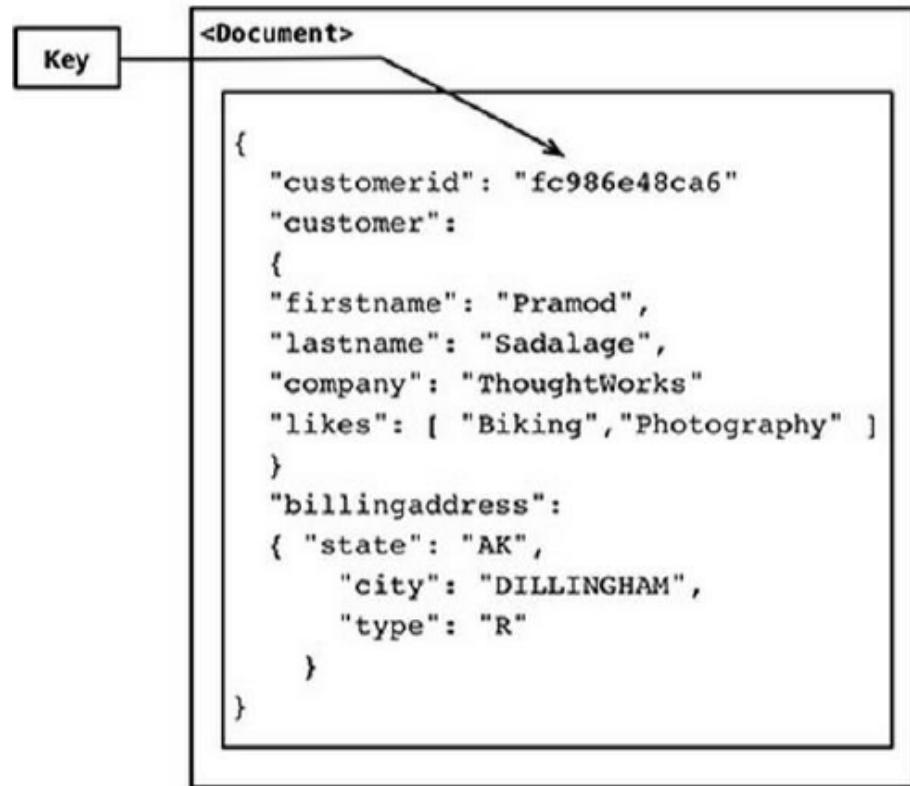
# NoSQL – Key–Value Databases

## ► Exemplos de Sistemas:

- Riak
- Redis
- Memcached
- BerkeleyDB
- HamsterDB
- Couchbase

# NoSQL – Document Databases

- ▶ O documento é o conceito base destas bases de dados
- ▶ O BD armazena e permite acessar os documentos
- ▶ Os documentos são armazenados em XML, JSON, BSON,...



# NoSQL – Document Databases

- ▶ Os documentos são auto-descritivos. São similares, mas não necessariamente iguais
- ▶ Os documentos são conjuntos de elementos etiquetados (tagged) e que podem ser pesquisados
- ▶ Cada documento tem um campo chave
- ▶ Podem ser criados índices para acelerar as pesquisas

# NoSQL – Document Databases

## ► Exemplos de Sistemas:

- MongoDB
- CouchDB
- Terrastore
- OrientDB
- RavenDB

# NoSQL – Column-Family Databases

- ▶ Cada bloco de armazenamento contém dados de uma coluna de uma tabela/registro/documento

PERSON TABLE					
row key	personal_data		demographic		...
PersonID	Name	Address	BirthDate	Gender	...
1	H. Houdini	Budapest, Hungary	1926-10-31	M	...
2	D. Copper	New Jersey, USA	1956-09-16	M	
3	Merlin	Stonehenge, England	1136-12-03	F	
...	...	...	...	...	
500,000,000	E. Cadillac	Nevada, USA	1964-01-07	M	

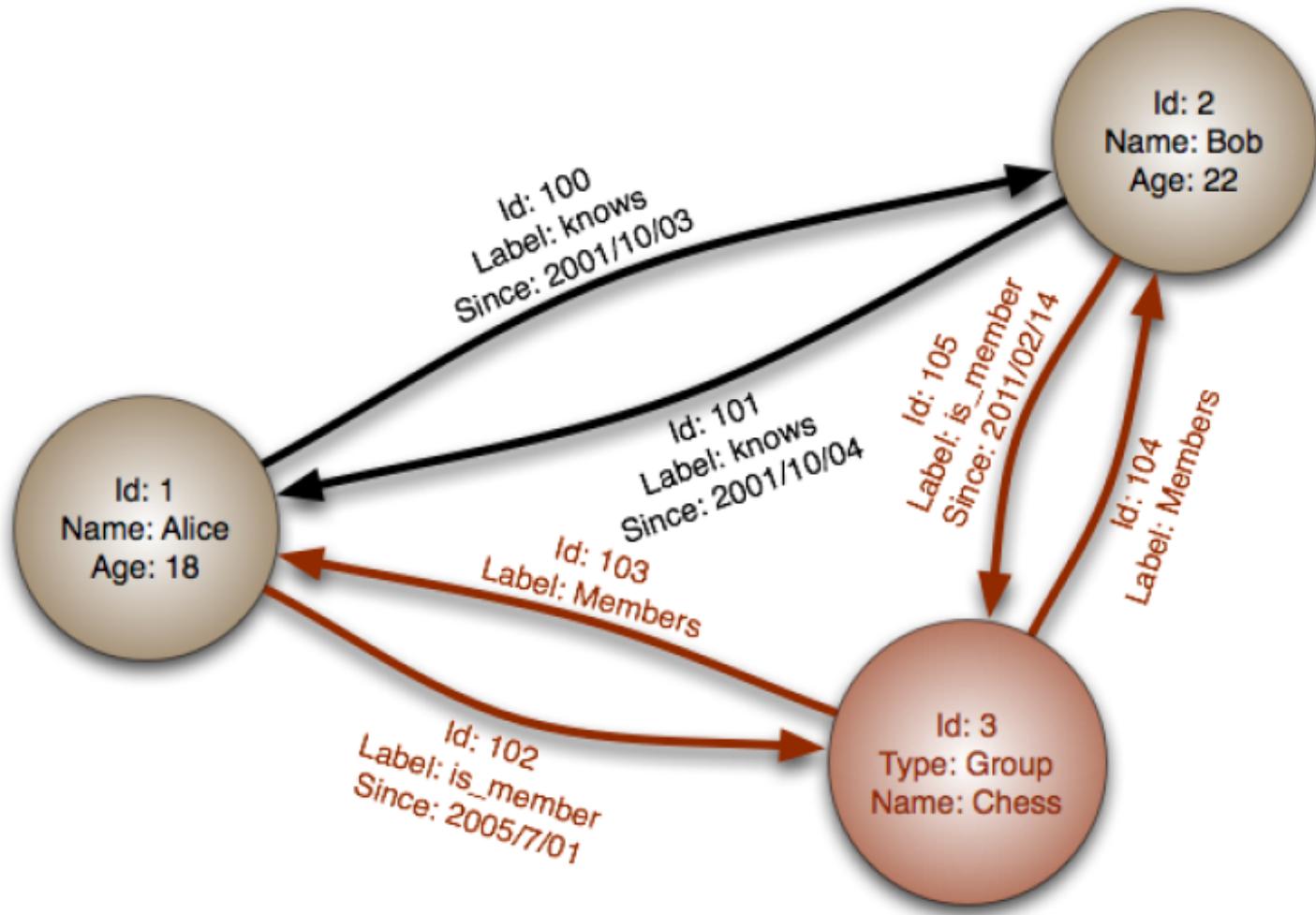
# NoSQL – Column Databases

- ▶ Mais eficiente se múltiplas linhas (ou documentos) são armazenados em conjunto. Assim, as atualizações podem ser agregadas
- ▶ Exemplos:
  - Cassandra
  - Hbase
  - Hypertable
  - Amazon DynamoDB

# NoSQL – Graph Databases

- ▶ Entidades são nós de um grafo
- ▶ Associações entre entidades são representadas por arestas
- ▶ Entidades e associações podem ter propriedades
- ▶ A organização do grafo permite que seja interpretado de diferentes formas, baseadas nas associações entre os nós

# NoSQL - Graph Databases



# NoSQL – Graph Databases

- ▶ Exemplos de Sistemas:

- Neo4J
- Infinite Graph
- OrientDB

# SQL vs NoSQL – Escalabilidade

- ▶ Para enfrentar o aumento no número de usuários e o aumento do volume de dados podem considerar-se duas hipóteses
- ▶ Scale-Up – Usar servidores cada vez maiores (solução centralizada)
- ▶ Scale-Out – Distribuir os dados e a computação por múltiplos servidores

# SQL vs NoSQL – Escalabilidade

- ▶ O modelo relacional por omissão escala utilizando servidores cada vez mais poderosos.
- ▶ Isto implica planejar, dimensionar, adquirir o servidor e, eventualmente, parar a aplicação para ser migrada (ScaleUp)

# SQL vs NoSQL – Escalabilidade

- ▶ Bases de dados **NoSQL** foram desenhadas para ser distribuídas e tolerantes a falhas
- ▶ São open-source
- ▶ Usam máquinas comuns (mais baratas)
- ▶ Se o volume de dados aumenta, acrescentam-se servidores

# NoSQL – Escalabilidade e Desempenho

- ▶ **Auto-Sharding**
- ▶ Uma base de dados NoSQL automaticamente divide os seus dados por vários servidores, sem parar a aplicação
- ▶ Servidores podem ser adicionados ou removidos
- ▶ Muitos sistemas NoSQL replicam os dados pelos servidores para garantir disponibilidade e tolerância a falhas

# NoSQL – Escalabilidade e Desempenho

- ▶ **Distributed Query Support**
  - Sistemas NoSQL fornecem suporte para queries distribuídas
- ▶ **Integrated Caching**
  - Para aumentar o desempenho, colocam os dados em memória de forma transparente para a aplicação