

# Performance e Otimização

**Capítulo 1. Bancos de Dados Distribuídos**

**PROF. GUSTAVO AGUILAR**

# Bancos de Dados Distribuídos

---

CAPÍTULO 1. AULA 1.1. FUNDAMENTOS DE SISTEMAS DISTRIBUÍDOS

PROF. GUSTAVO AGUILAR

# Nesta Aula



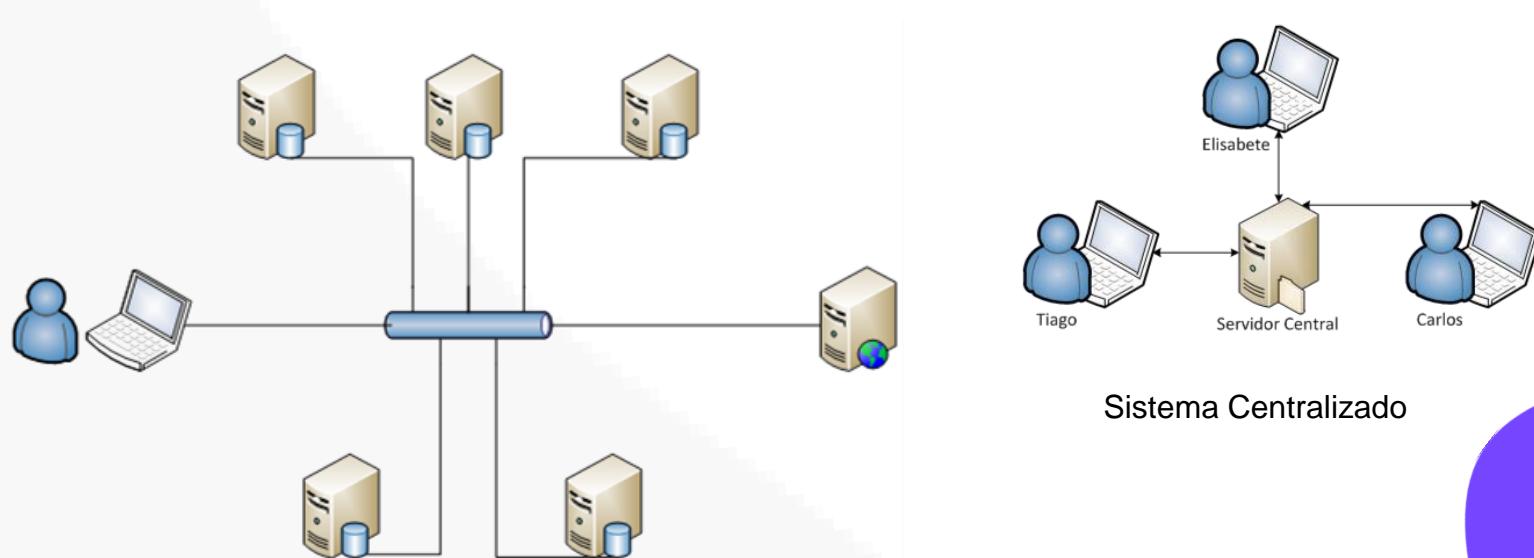
- ❑ O Que É Sistema Distribuído?
- ❑ Macroevolução da Computação.
- ❑ Macroevolução de Sistemas Distribuídos.
- ❑ Características de Sistemas Distribuídos.

# O Que É Sistema Distribuído?

IGTI

- “Um sistema constituído por um conjunto de computadores independentes, visto pelos utilizadores do sistema como sendo um sistema coerente e único.”

(Tanenbaum - Sistemas Distribuídos: Princípios e Paradigmas)



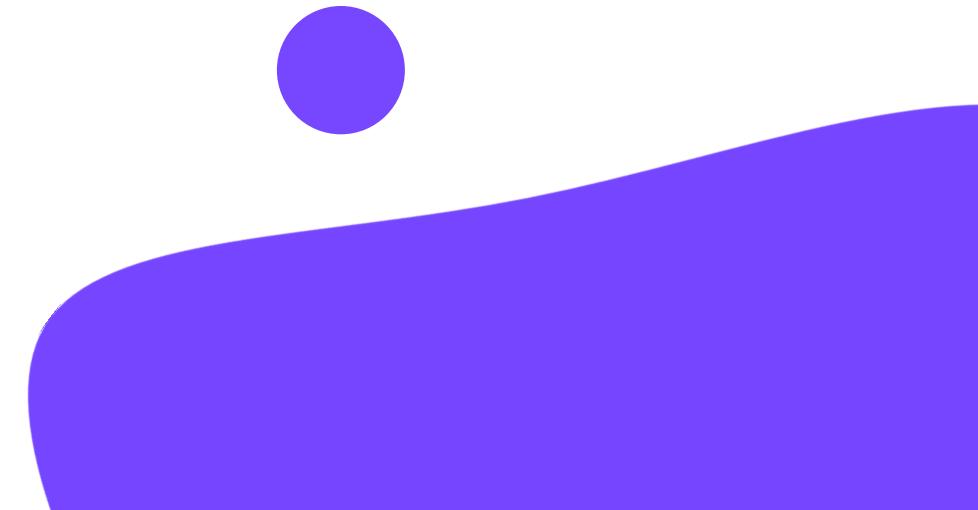
Sistema Centralizado

# O Que É Sistema Distribuído?

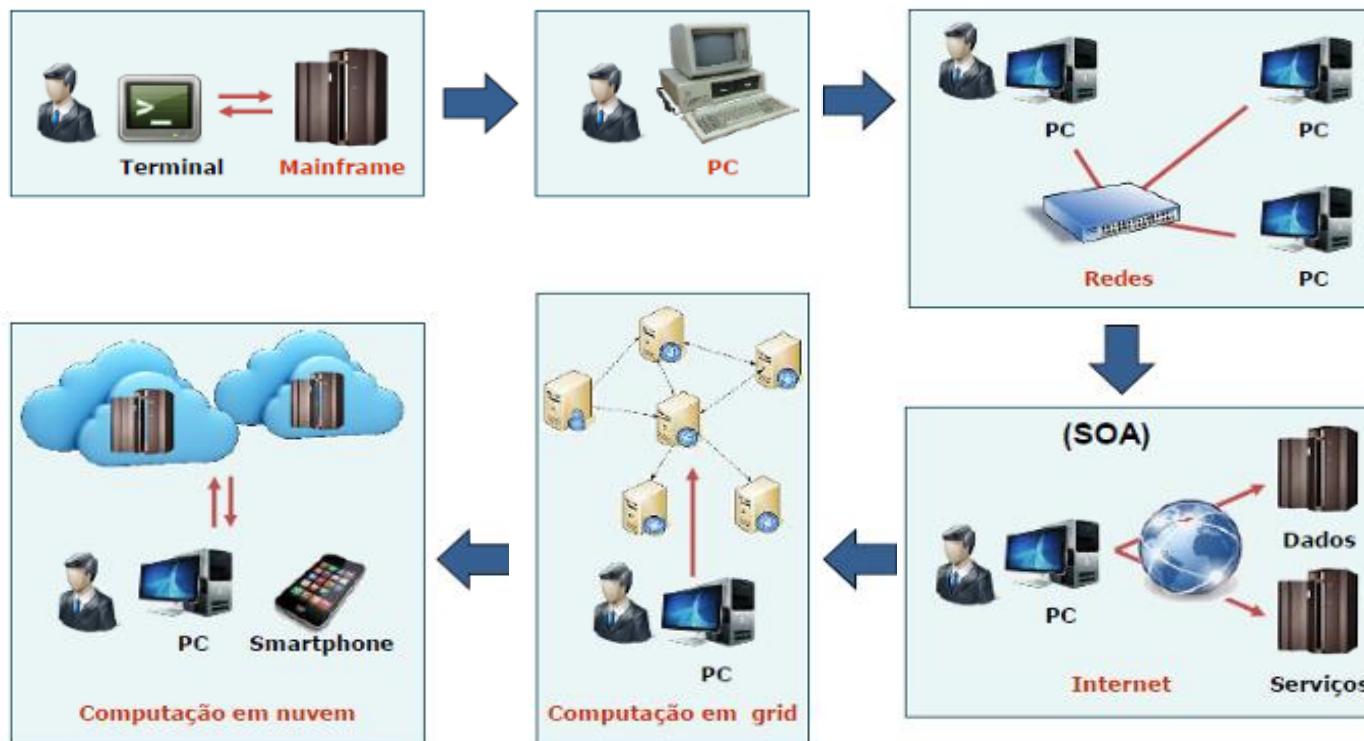
IGTI

- “Um sistema distribuído é aquele no qual os componentes localizados em computadores **interligados em rede** se comunicam e coordenam suas ações apenas passando mensagens”.

(Coulouris – Sistemas Distribuídos: Conceitos e Projetos)



# Macroevolução da Computação



# Macroevolução de Sistemas Distribuídos



IoT (Internet das Coisas) x IA (Inteligência Artificial) x Blockchain x Big Data x 5G x SDN (Rede Definida por Software)

FrontEnd x API x Microserviço x BackEnd x SOA x ITCore

Cliente x Solução  
FrontEnd x Integração x BackEnd x Banco

Cliente (N camadas)  
Interface x Web x Middleware x Banco

Cliente x Web (4 camadas)  
Browser x Web x Aplicação x Banco

Cliente x Servidor (3 camadas)  
Estação x Aplicação x Banco

Cliente x Servidor (2 camadas)  
Estação x Banco

Mainframe x Terminal

Microserviço e API  
Digital, DevOps, Bimodal, Ágil, CI/CD, IaC

Web Services, SOA (Arquitetura Orientada a Serviços), ESB (Enterprise Service Bus)

Middleware Server, Message Queue, BI/DW, ETL,  
Report Server e Workflow

RMI (Remote Método Invocation) J2EE  
RPC (Remote Procedure Call) COM/DCOM

Java e Microsoft (Virtual Machine)  
Corba (Transações Distribuídas)

TMVE e MVC (Modelo x Visão x Controle)  
(Apresentação e Lógica de Negócio)

Tela x Programa x Sub-Rotina

Cloud (SaaS, IaaS, PaaS)  
NFV (Virtualização das Funções da Rede)

DR (Disaster Recovery), APN (Access Point Name) e GLB (Global Load Balance)

WPAN e PAN (Personal Area Network)  
WiFi e Bluetooth

MPLS, VPN, Rede Unicast e Multicast  
Internet, Extranet e Intranet

WAN (Wide Area Network)  
Balanceado x Cluster

LAN e MAN e SAN  
Cluster

LAN

Arquitetura

Aplicação

Rede

# Características de Sist. Distribuídos



- Conectividade
- Disponibilidade
- Desempenho
- Escalabilidade
- Transparência
- Interoperabilidade
- Segurança
- Tolerância a falhas
- Usabilidade
- Heterogeneidade

# Características de Sist. Distribuídos

IGTI

- **Conectividade**

- Característica voltada para a capacidade do sistema se manter conectado a recursos disponíveis.
- **Desafio:** computação móvel + computação ubíqua



# Características de Sist. Distribuídos



## ▪ Disponibilidade

- Maximizar o uptime → reduzir o downtime.

Disponibilidade (%)	Downtime/ano	Downtime/mês
95%	18 dias 6:00:00	1 dia 12:00:00
96%	14 dias 14:24:00	1 dia 4:48:00
97%	10 dias 22:48:00	0 dias 21:36:00
98%	7 dias 7:12:00	0 dias 14:24:00
99%	3 dias 15:36:00	0 dias 7:12:00
99,90%	0 dias 8:45:35.99	0 dias 0:43:11.99
99,99%	0 dias 0:52:33.60	0 dias 0:04:19.20
99,999%	0 dias 0:05:15.36	0 dias 0:00:25.92

# Características de Sist. Distribuídos

IGTI

- **Desempenho**

- Tempo que o ecossistema leva para processar uma tarefa.



# Características de Sist. Distribuídos



## ▪ Escalabilidade

- Capacidade de escalar o ambiente ou seja, de adicionar mais recursos, de forma transparente para o usuário e a aplicação;
- Poder de processamento (CPU)
- Cache (memória RAM)
- Capacidade de armazenamento (disco)
- Velocidade de transmissão (network)
- Nós de processamento (worker nodes)

## ▪ Escalabilidade X Elasticidade

# Características de Sist. Distribuídos

IGTI

- **Escalabilidade**

- **Vertical:** adicionar CPU, RAM, disco nos integrantes do ambiente.



# Características de Sist. Distribuídos

IGTI

- **Escalabilidade**

- **Horizontal:** adicionar novos integrantes ao ambiente.



# Características de Sist. Distribuídos



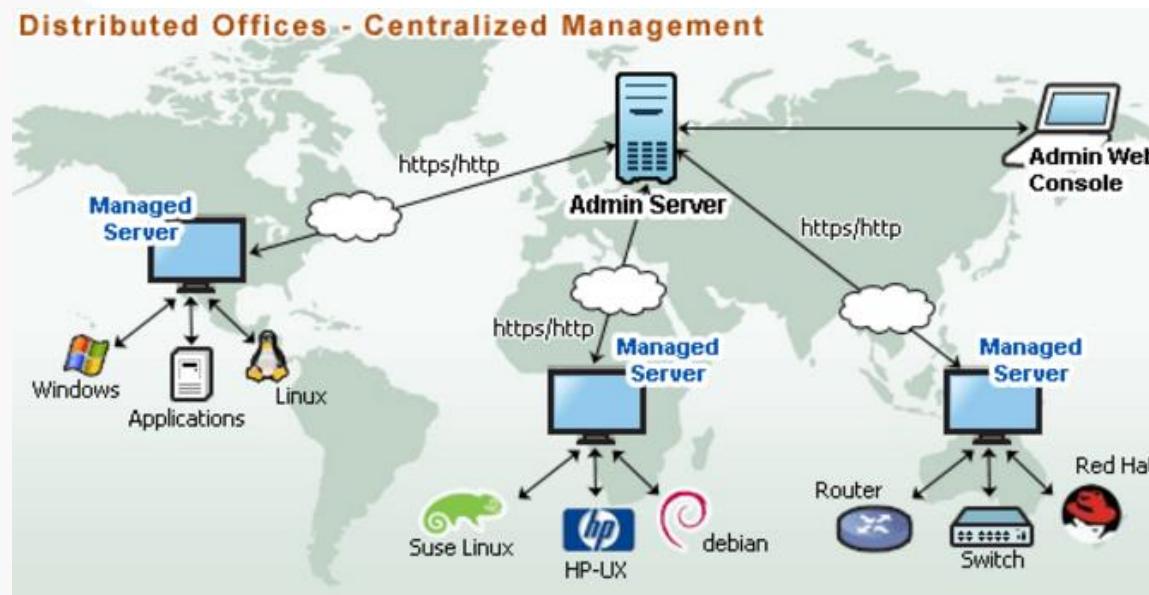
- **Transparência**

- Ocultação, para o usuário final ou para o desenvolvedor, **da separação dos componentes** em um sistema distribuído;
- Sistema percebido como um todo, em vez de como uma coleção de componentes independentes e isolados.
- De acesso
- De localização
- De falha
- De replicação
- De persistência
- De concorrência
- De migração e realocação

# Características de Sist. Distribuídos

- **Transparência**

- **De acesso:** esconde os detalhes dos protocolos e configurações de rede que controlam a comunicação entre as diversas máquinas, ocultando diferenças entre arquiteturas de máquinas.



# Características de Sist. Distribuídos

IGTI

- **Transparência**

- **De localização:** esconde a localização dos recursos no sistema distribuído, de forma que os usuários não são capazes de dizer a localização física do recurso.



# Características de Sist. Distribuídos



- **Transparência**

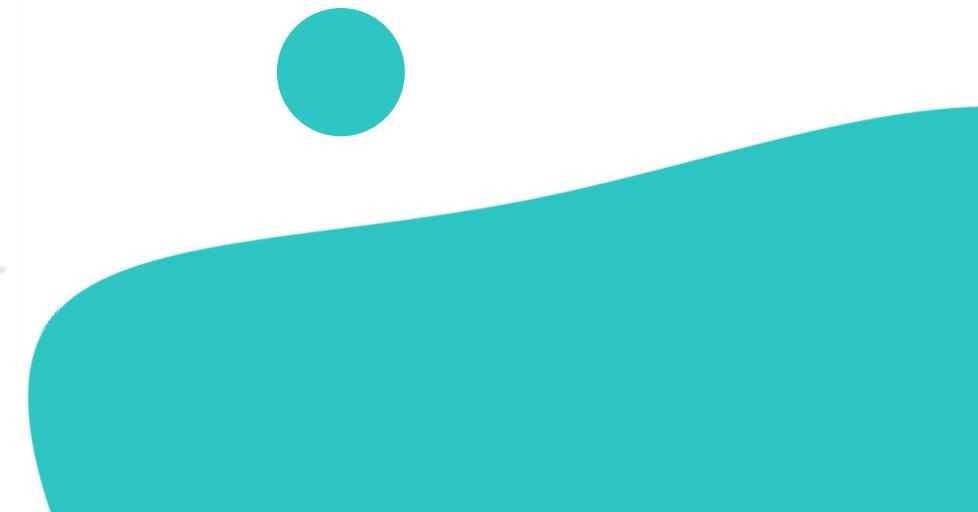
- **De replicação:**

- Esconde o fato de que múltiplas cópias do mesmo recurso podem estar disponíveis no sistema;
    - Permite que várias instâncias de recursos sejam usadas para aumentar a confiabilidade e o desempenho;
    - Deve mascarar o conhecimento das réplicas por parte dos usuários;
    - Implica na transparência de localização.

# Características de Sist. Distribuídos

IGTI

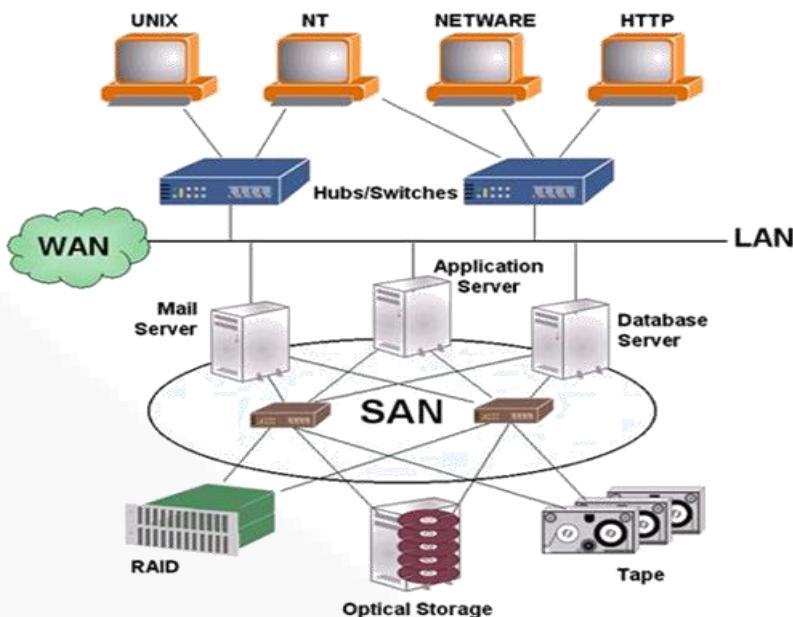
- Transparência
  - De replicação



# Características de Sist. Distribuídos

- **Transparência**

- **De falha:** falhas no sistema não são percebidas pelo usuário.
  - Provê alta disponibilidade.

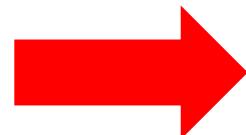


# Características de Sist. Distribuídos

IGTI

- **Transparência**

- **De persistência:** meio de armazenamento e detalhes físicos da persistências e durabilidade dos dados devem ser transparentes para o usuário.



Impacta na experiência  
do usuário

IBM 305 RAMAC (1956)  
5 MB  
14x8 polegadas

# Características de Sist. Distribuídos

- **Transparência**

- **De concorrência:** qualquer recurso compartilhado em um sistema distribuído deve ter garantido o correto funcionamento em um ambiente concorrente.



# Características de Sist. Distribuídos



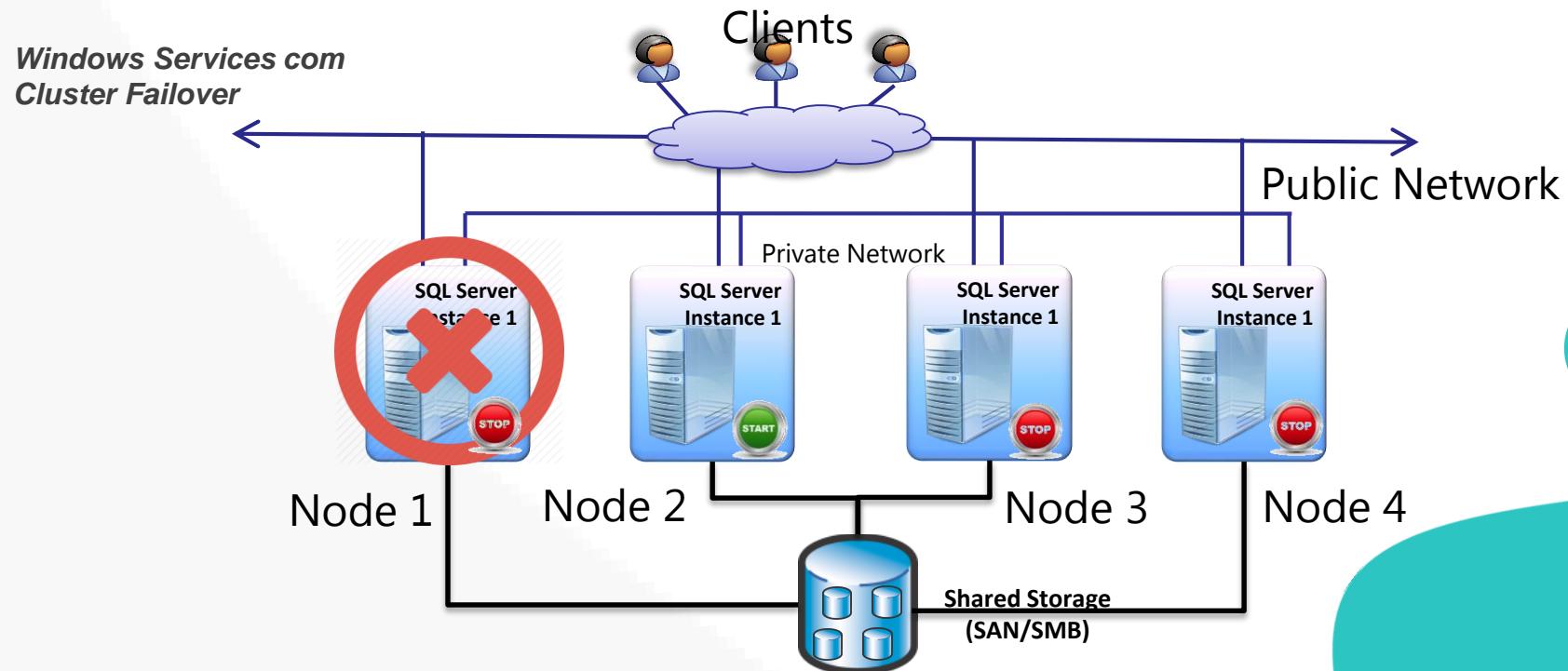
- **Transparência**

- **De migração e realocação:**

- Escondem a movimentação de recursos em um sistema distribuído, mascarando a movimentação de um objeto de um ponto a outro no sistema;
    - Recursos podem migrar de uma localidade para outra, por questões de desempenho, segurança, etc.;
    - Deve ser feita de forma automática pelo sistema;
    - Deve manter o nome do recurso que o usuário conhece;
    - Deve garantir a continuidade de comunicação.

# Características de Sist. Distribuídos

- Transparência
  - De migração e realocação



# Características de Sist. Distribuídos



## ▪ Interoperabilidade

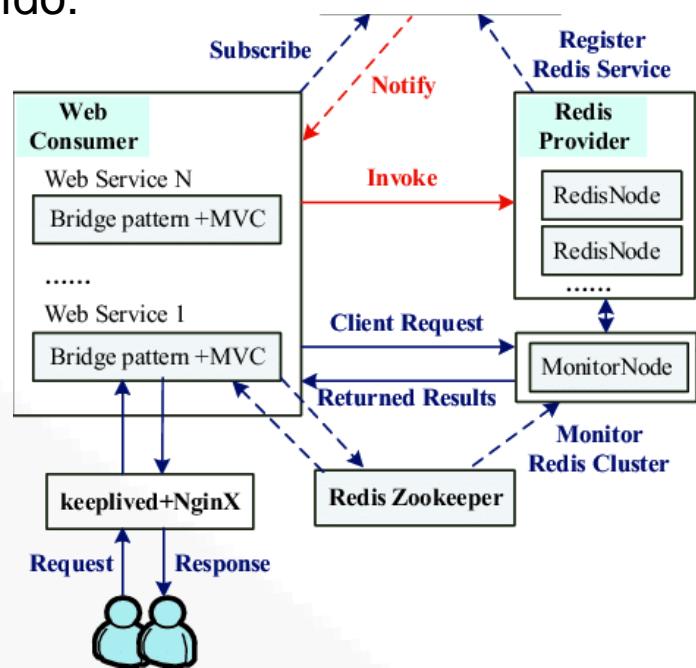
- Demonstra a capacidade que o sistema possui de interagir e comunicar com outros sistemas/mecanismos de forma transparente e simples;
- Abrange a capacidade que o ecossistema tem de integrar tecnologias heterogêneas e apresentá-las de forma homogênea para utilização.

# Características de Sist. Distribuídos

IGTI

- **Segurança**

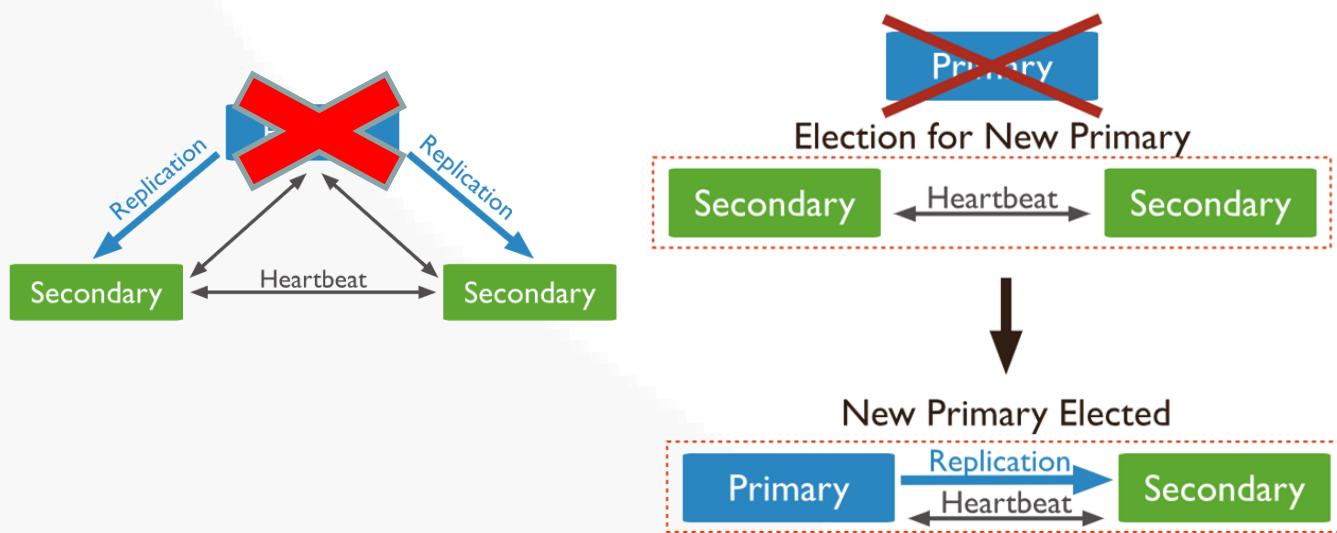
- Garantir acesso a seus recursos apenas a usuários/sistemas devidamente identificados e autorizados, em qualquer ponto do sistema distribuído.



# Características de Sist. Distribuídos

- **Tolerância a Falhas**

- Propriedade que permite que sistemas continuem a operar adequadamente, mesmo após falha(s) em algum de seus componentes;
- Tolerância a Falhas X Alta Disponibilidade;
- Tolerância a Falhas X Transparência de Falha.



# Características de Sist. Distribuídos



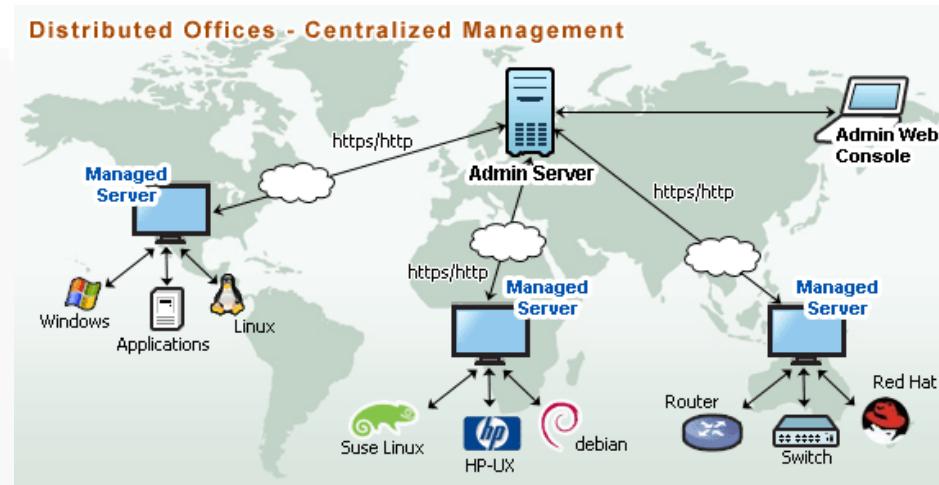
## ▪ Usabilidade

- É a capacidade que um sistema dispõe de tornar a sua utilização simples e objetiva;
- Em sistemas distribuídos, essa definição também contempla a transparência de acesso ao usuário, abstraindo-o do conhecimento do ecossistema.

# Características de Sist. Distribuídos

## ▪ Heterogeneidade

- Sistemas distribuídos normalmente são construídos a partir de uma variedade de redes, sistemas operacionais, hardwares e linguagens de programação distintas.
- Protocolos de comunicação e middlewares para mascarar essa diferença, tornando o sistema homogêneo



# Próxima Aula



- ❑ Introdução à Bancos de Dados Distribuídos.

# Bancos de Dados Distribuídos

---

CAPÍTULO 1. AULA 1.2. INTRODUÇÃO À BANCOS DE DADOS DISTRIBUÍDOS

PROF. GUSTAVO AGUILAR

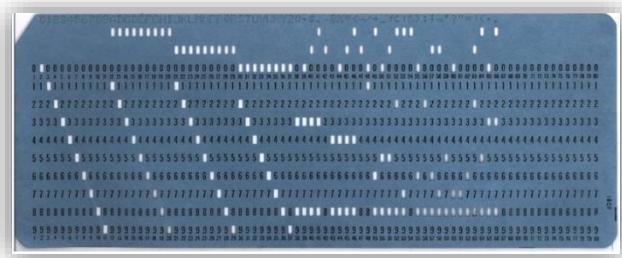
# Nesta Aula



- Revisão de Conceitos.
- Banco de Dados Distribuídos.
- SGBDD.
- SGBD Centralizado.

# Banco de Dados

igit



Generic,Storage Type,Hierarchy Type,Attributes Header,comment,bso data storage,aso data storage,  
two pass calculation,aso dimension formula,consolidation type,uda,parent,alias:Default,alias:English  
Customers,SPARSE,STORED,,LABELONLY,STOREDATA,N,,,UDA,alias:Default,alias:English  
NoCustomer,SPARSE,Disabled,,,StoreData,StoreData,N,,+,No Customer,No Customer  
AllCustomers,SPARSE,Disabled,,,StoreData,StoreData,N,,+,All Customers,All Customers  
Big Box,SPARSE,,,StoreData,StoreData,N,,+,All Customers,,  
BB100,SPARSE,,,StoreData,StoreData,N,,+,Big Box,O Mart,O Mart  
BB200,SPARSE,,,StoreData,StoreData,N,,+,Big Box,Bike Depot,Bike Depot  
BB300,SPARSE,,,StoreData,StoreData,N,,+,Big Box,Mountain Adventures,Mountain Adventures  
Specialty Retailers,SPARSE,,,StoreData,StoreData,N,,+,All Customers,,  
SR100,SPARSE,,,StoreData,StoreData,N,,+,Specialty Retailers,Bobs Bikes,Bobs Bikes  
SR200,SPARSE,,,StoreData,StoreData,N,,+,Specialty Retailers,Rose Town Bikes,Rose Town Bikes  
SR300,SPARSE,,,StoreData,StoreData,N,,+,Specialty Retailers,The Cyclery,The Cyclery  
Webstore,SPARSE,,,StoreData,StoreData,N,,+,Allcustomers,,

# Banco de Dados



“É uma coleção de dados inter-relacionados, representando informações sobre um domínio específico.”

(KORTH, 1994, p.1)

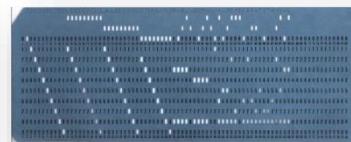


“É uma coleção de dados relacionados.”

(ELMASRI; NAVATHE, 2011, p.4)

**Banco de Dados Digital**

Generic,Storage Type,Hierarchy Type,Attributes Header,comment,bso data storage,aso data storage,  
two pass calculation,aso dimension formula,consolidation type,uda,parent,alias:Default,alias:English  
Customers,SPARSE,STORED,,,LABELONLY,STORED,TA,,NDA,,alias:Default,alias:English  
NoCustomer,SPARSE,Disabled,,,StoreData,StoreData,N,,+,No Customer,No Customer  
AllCustomers,SPARSE,Disabled,,,StoreData,StoreData,N,,+,All Customers,All Customers  
Big Box,SPARSE,,,StoreData,StoreData,,+,AllCustomers,  
BB100,SPARSE,,,StoreData,StoreData,N,,+,Big Box,Q Mart,Q Mart  
BB200,SPARSE,,,StoreData,StoreData,N,,+,Big Box,Bike Depot,Bike Depot  
BB300,SPARSE,,,StoreData,StoreData,N,,+,Big Box,Mountain Adventures,Mountain Adventures  
Specialty Retailers,SPARSE,,,StoreData,StoreData,N,,+,AllCustomers,  
SR100,SPARSE,,,StoreData,StoreData,N,,+,Specialty Retailers,Bobs Bikes,Bobs Bikes  
SR200,SPARSE,,,StoreData,StoreData,N,,+,Specialty Retailers,Rose Town Bikes,Rose Town Bikes  
SR300,SPARSE,,,StoreData,StoreData,N,,+,Specialty Retailers,The Cyclery,The Cyclery  
Webstore,SPARSE,,,StoreData,StoreData,N,,+,AllCustomers,,



**Banco de Dados Manual**



# Sistema Ger. de Banco de Dados

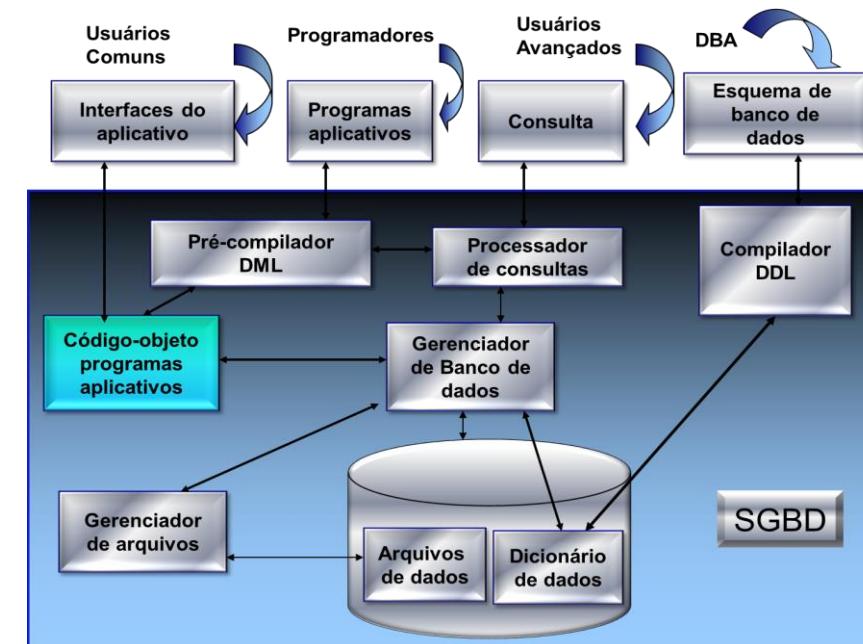


“É uma coleção de programas que permite aos usuários criar e manter um **banco de dados**.”



“Um sistema de software de propósito geral que facilita os processos de definição construção, manipulação e compartilhamento de **bancos de dados** entre vários usuários e aplicações.”

(ELMASRI; NAVATHE, 2011, p.4)



# Sistema Ger. de Banco de Dados

IGTI

- ✓ Sistemas Gerenciadores de Bancos de Dados Relacional (SGBDR)



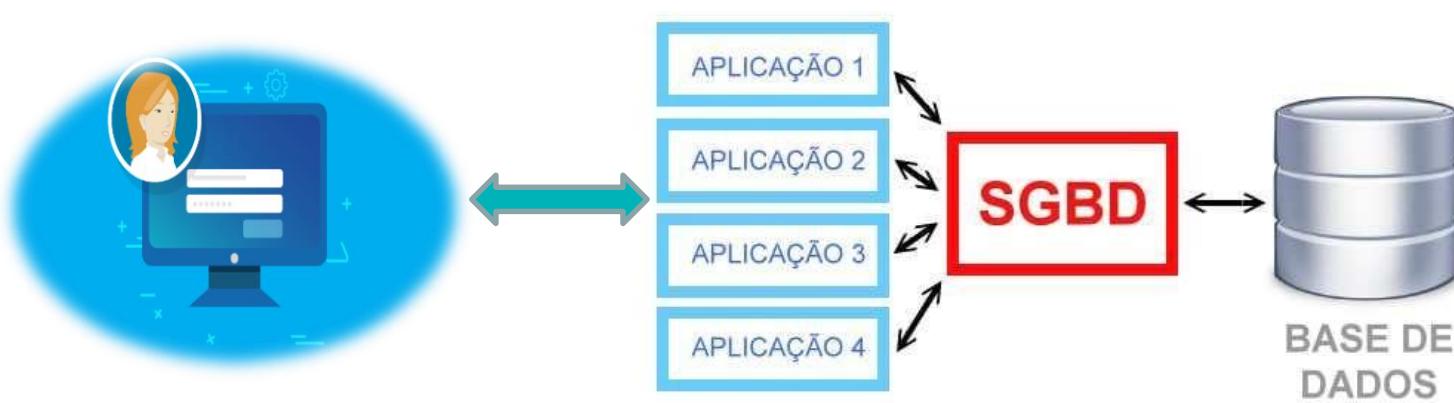
- ✓ Sistemas Gerenciadores de Bancos de Dados NOSQL



# Sistema de Banco de Dados



Ecossistema formado por um SGBD, o banco de dados e os programas que permitem à um usuário ou máquina, manipular e consultar dados.



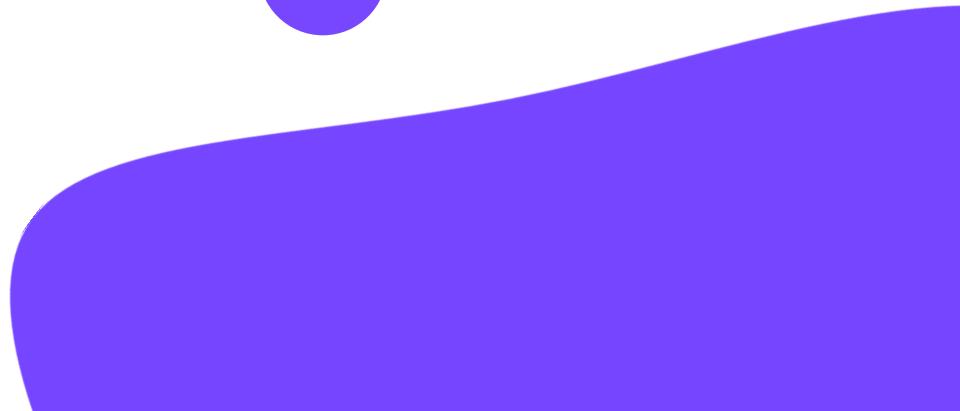
# Banco de Dados Distribuídos



“Uma coleção de múltiplos bancos de dados logicamente inter-relacionados e distribuídos por uma rede de computadores”.  
(ELMASRI; NAVATHE)

Pode ser:

- Relacional
- NoSQL
- NewSQL
- ...

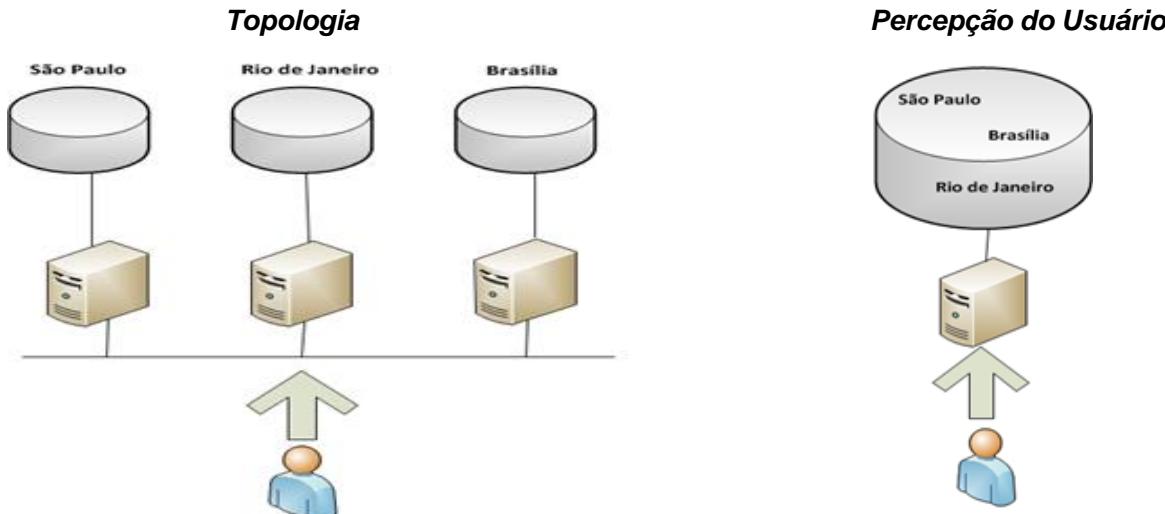


# SGBDD

Sistema Gerenciador de Banco de Dados Distribuídos



“Sistema de software que gerencia um banco de dados distribuído enquanto torna a distribuição transparente para o usuário.  
”  
(ELMASRI;NAVATHE, 2011, p.579)



# SGBDD

IGTI

- Exemplos de SGBDD



# SGBDD

- Tipos de Sistemas de Bancos de Dados Distribuídos:

## Homogêneo



- SGBDs com software idêntico:
  - Mesmo fornecedor;
  - Mesma versão;
  - Mesma edição.
- Usuários / clients usam o mesmo software para acessar os BDDs.

## Heterogêneo



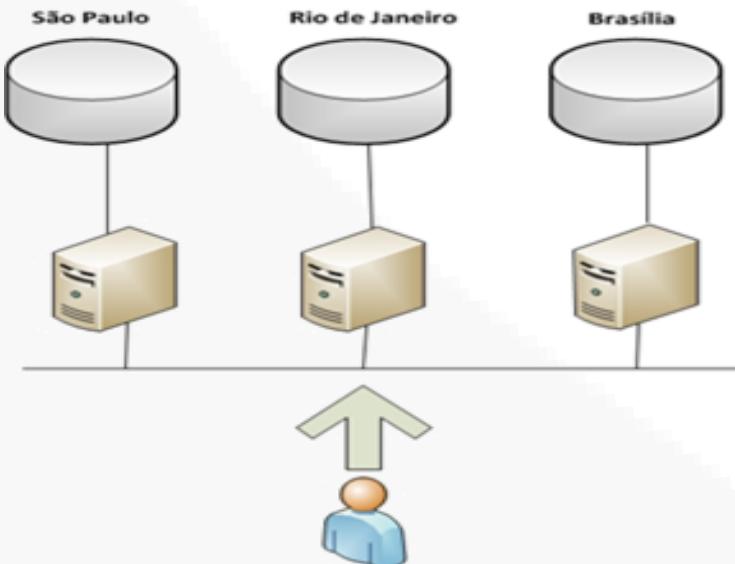
- SGBDs com software diferente:
  - Fornecedor diferente; ou
  - Versões diferentes; ou
  - Edições diferentes.
- Usuários / clients precisam usar softwares diferentes para acessar os BDDs.

# 12 Regras de um SGBDD



Em 1987, Christopher J. Date enumerou 12 “regras” básicas que um Sistema Gerenciador de Banco de Dados Distribuídos completo deveria possuir:

- Autonomia Local:** cada nó deve prover seus próprios mecanismos de segurança, bloqueio, acesso, integridade e recuperação após uma falha.



Na prática, se resume a cada nó ser uma instância do SGBDD, cada uma mantendo o controle sobre seus próprios bancos de dados.

# 12 Regras de um SGBDD

- Descentralização:** não dependência de um nó central, o que acarretaria em um único ponto de falha.
  - Se um nó ficar indisponível, a continuidade das operações no ambiente deve ser garantida.



- Operação Contínua:** operações de replicação e distribuição dos dados, backup e recuperação devem ser suportadas de forma on-line.
  - Rápidas o bastante para não afetarem o funcionamento do sistema.

# 12 Regras de um SGBDD

- Independência de Hardware:** as operações em bancos de dados distribuídos não devem estar condicionadas à recursos físicos ou lógicos de hardware.
- Independência de Sistema Operacional:** as operações em bancos de dados distribuídos não devem estar condicionadas à recursos de um sistema operacional específico.
- Independência de Rede:** deve executar independentemente do protocolo de comunicação e da topologia de rede usada para interligar os nós.
- Independência de SGBD:** um SGBDD ideal deve possuir capacidade(s) para se comunicar com outros SGBDs → interoperabilidade.
- Processamento Distribuído de Consultas:** experiência do usuário, em termos de tempo de resposta, deve ser independente do local de disparo.
- Gerenciamento de Transações Distribuídas.**

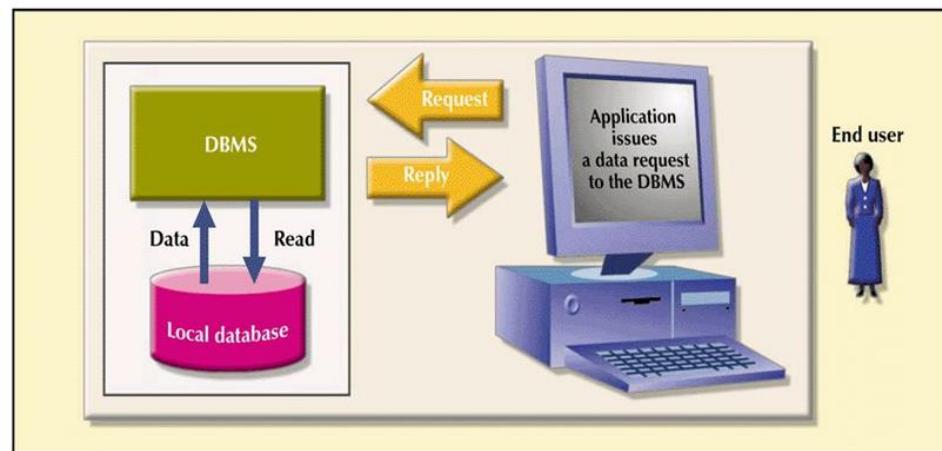
# 12 Regras de um SGBDD



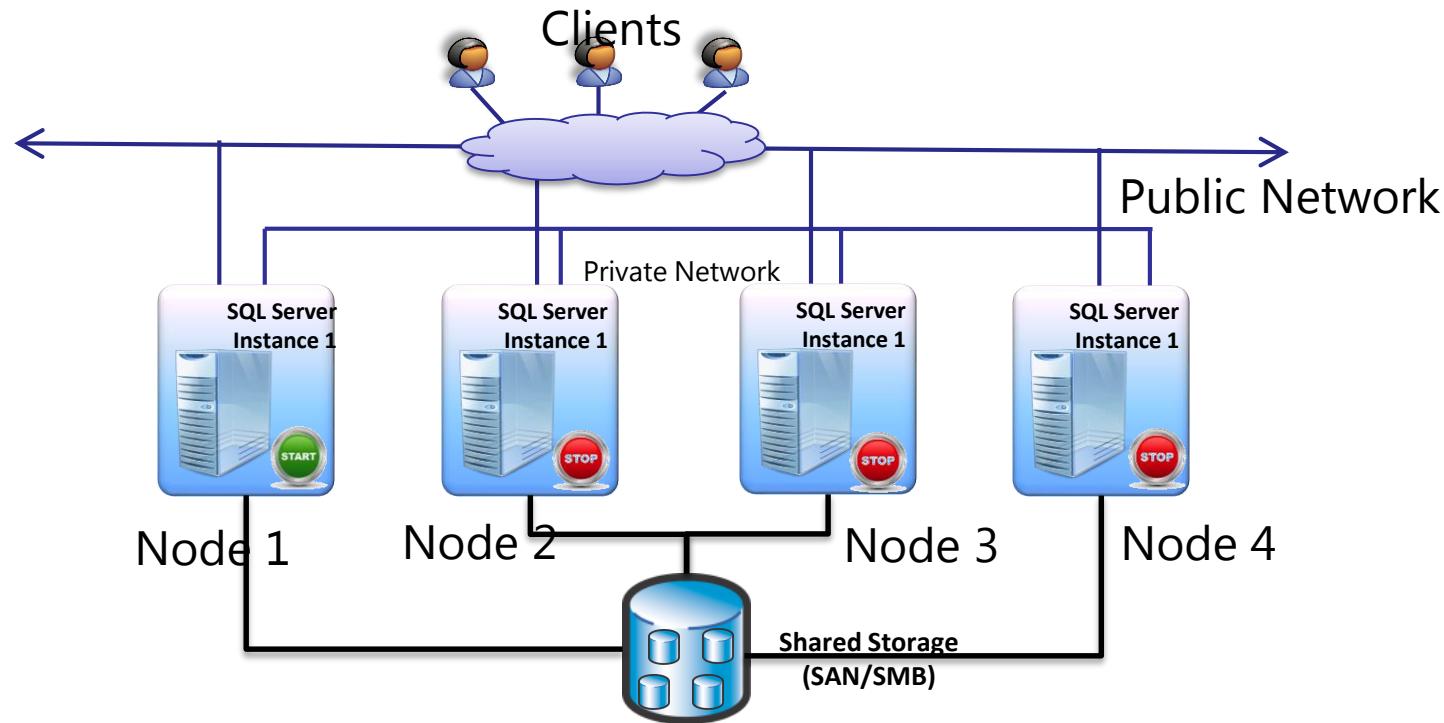
- ✓ **Transparência / Independência de Localização:** os usuários do sistema não necessitam saber a localização do armazenamento dos dados.
- ✓ **Independência de Replicação:** dados podem estar replicados em vários nós da rede, sem interferir na visão única do usuário.
  - As réplicas de dados devem ser mantidas sincronizadas automaticamente pelo SGBDD.
- ✓ **Independência de Fragmentação:** banco de dados ou tabelas podem estar divididos em fragmentos, localizados fisicamente em diferentes nós, de forma transparente para o usuário.

# SGBD Centralizado

- Front-end único;
- Gerenciamento de dados centralizado;
- Processamento centralizado de consultas e transações;
- Único ponto de falha;
- Componentes físicos únicos ou compartilhados (Cluster);
- Escalabilidade apenas vertical.



# SGBD Centralizado

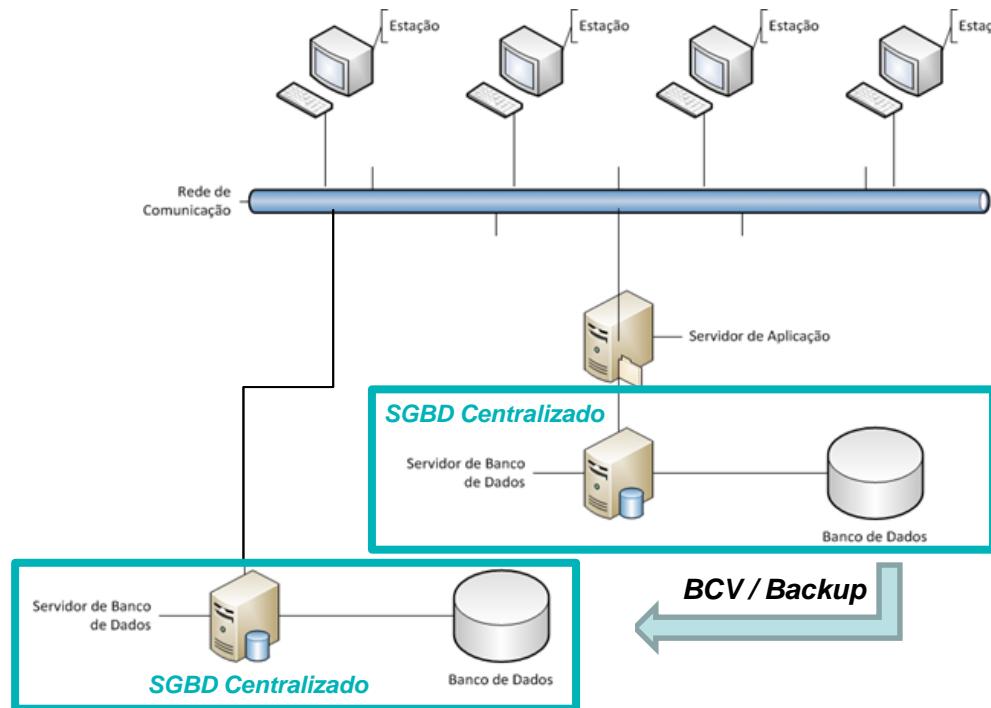


*SQL Server Failover Cluster*

# O Que Não é SGBDD



Um sistema gerenciador de banco de dados que não tem a capacidade nativa de distribuir ou replicar o banco de dados, não pode ser considerado um sistema gerenciador de banco de dados distribuído, apesar do banco de dados do sistema poder estar **duplicado fisicamente**.



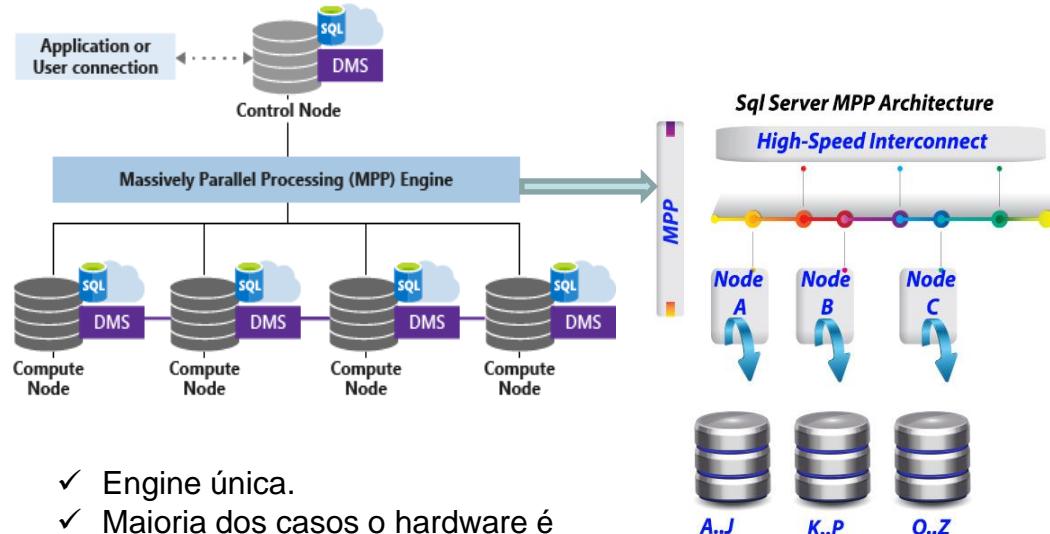
Sistema com o Banco de Dados Duplicado Fisicamente

# O Que Não é SGBDD

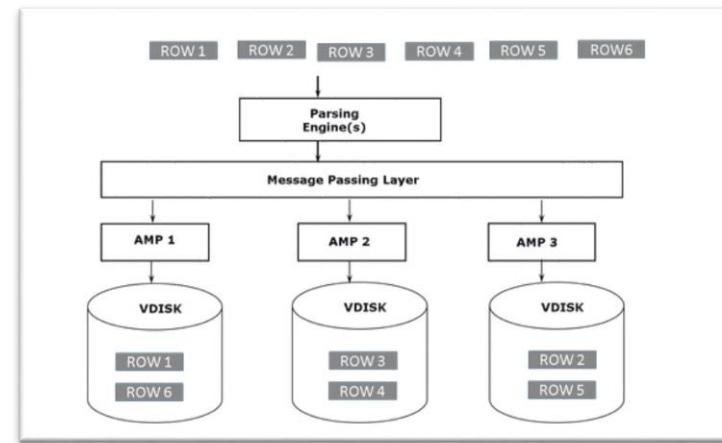
IGTI



Banco de Dados Paralelo (MPP), não é sinônimo de BD distribuído!



teradata.



- ✓ Engine única.
- ✓ Maioria dos casos o hardware é único (nodes são lógicos).
- ✓ Rede: barramento entre os nós.

# Por Que NÃO Sistemas de Bancos de Dados Distribuídos?

IGTI



# Próxima Aula



- Técnicas de Distribuição de Dados.

# Bancos de Dados Distribuídos

---

CAPÍTULO 1. AULA 1.3. TÉCNICAS DE DISTRIBUIÇÃO DE DADOS

PROF. GUSTAVO AGUILAR

# Nesta Aula



- Replicação de Dados.
- Particionamento de Dados.

# Replicação de Dados

- Cópias (réplicas) sendo armazenadas em mais de um local;
- Não são backups tradicionais (Backup Database / Dump / Export);

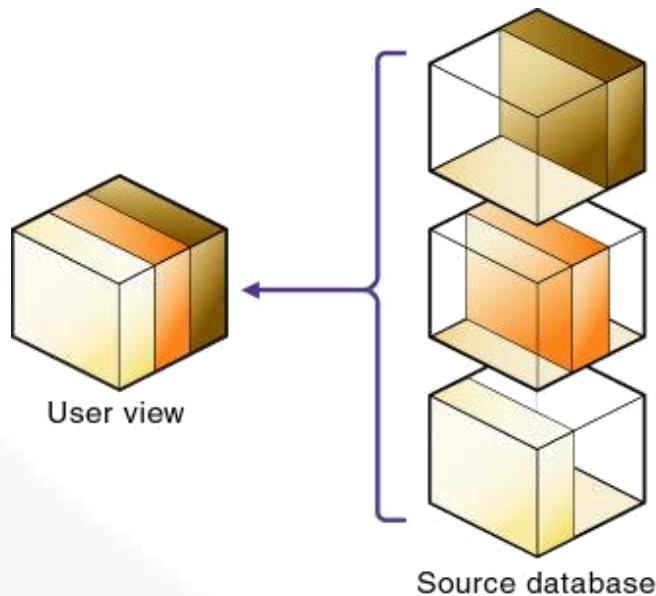


# Replicação de Dados

- Muito utilizada para ambientes de Disaster Recovery (DR);
- Utilizadas também para separação de workloads distintos;
- SGBDD é quem gerencia a replicação
  - ✓ Disponibilidade;
  - ✓ Consistência;
  - ✓ Integridade;
  - ✓ Confiabilidade dos dados.
- Transparência da replicação e da localização dos dados.
- Não se considera BDs replicados por softwares de terceiros.

# Particionamento de Dados

- Também conhecida como **fragmentação (sharding)**;



# Particionamento de Dados

- Muito utilizada em ambientes que requerem escalabilidade horizontal;
- Usa-se um hash ou faixa de valor para a distribuição;
- SGBDD é quem gerencia a distribuição
  - ✓ Disponibilidade, consistência, integridade e confiabilidade dos dados;
  - ✓ Transparência da distribuição e da localização dos dados;
  - ✓ Resolução de conflitos.
- Dados particionados pela aplicação + SGBD centralizado
  - ✓ Esses são considerados sistemas de banco de dados paralelos (MPP);
  - ✓ Tabelas particionadas em um mesmo SGBD não são SGBDDs.

# Próxima Aula



- ❑ Tipos de Replicação de Dados.

# Bancos de Dados Distribuídos

---

CAPÍTULO 1. AULA 1.4. TIPOS DE REPLICAÇÃO DE DADOS

PROF. GUSTAVO AGUILAR

# Nesta Aula

- Replicação Assíncrona.
- Replicação Síncrona.
- Replicação Hierárquica.
- Replicação Peer-To-Peer.
- Opções para Replicação de Dados.

# Tipos de Replicação de Dados



- **Quanto à Consistência:**

- ✓ Assíncrona

- ✓ Síncrona

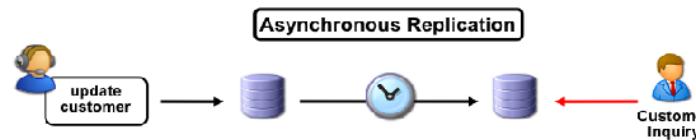
- **Quanto à Topologia:**

- ✓ Hierárquica

- ✓ Peer-To-Peer

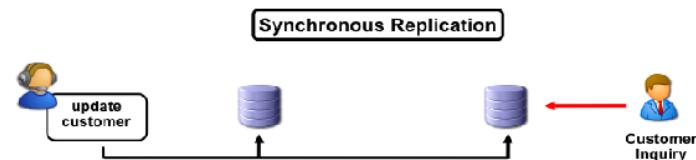
# Replicação Assíncrona

- Transação concluída em um nó e replicada para os demais:
  - ✓ Nó de origem da transação não aguarda a confirmação dos demais nós do ambiente;
  - ✓ Nó de origem garante a persistência da transação ocorrida nele;
  - ✓ SGBDD garante a persistência da transação distribuída em todos os nós.
- Mais performática para o usuário;
- Consistência eventual:
  - Pode ocorrer atraso na replicação.



# Replicação Síncrona

- Transação só é concluída após todos os nós confirmarem o commit:
  - ✓ Nó de origem da transação aguarda a confirmação dos demais nós do ambiente.
- Menos performática para o usuário;
- Consistência total:
  - ✓ Em qualquer momento, o usuário enxerga o mesmo conjunto de dados em qualquer um dos nós.



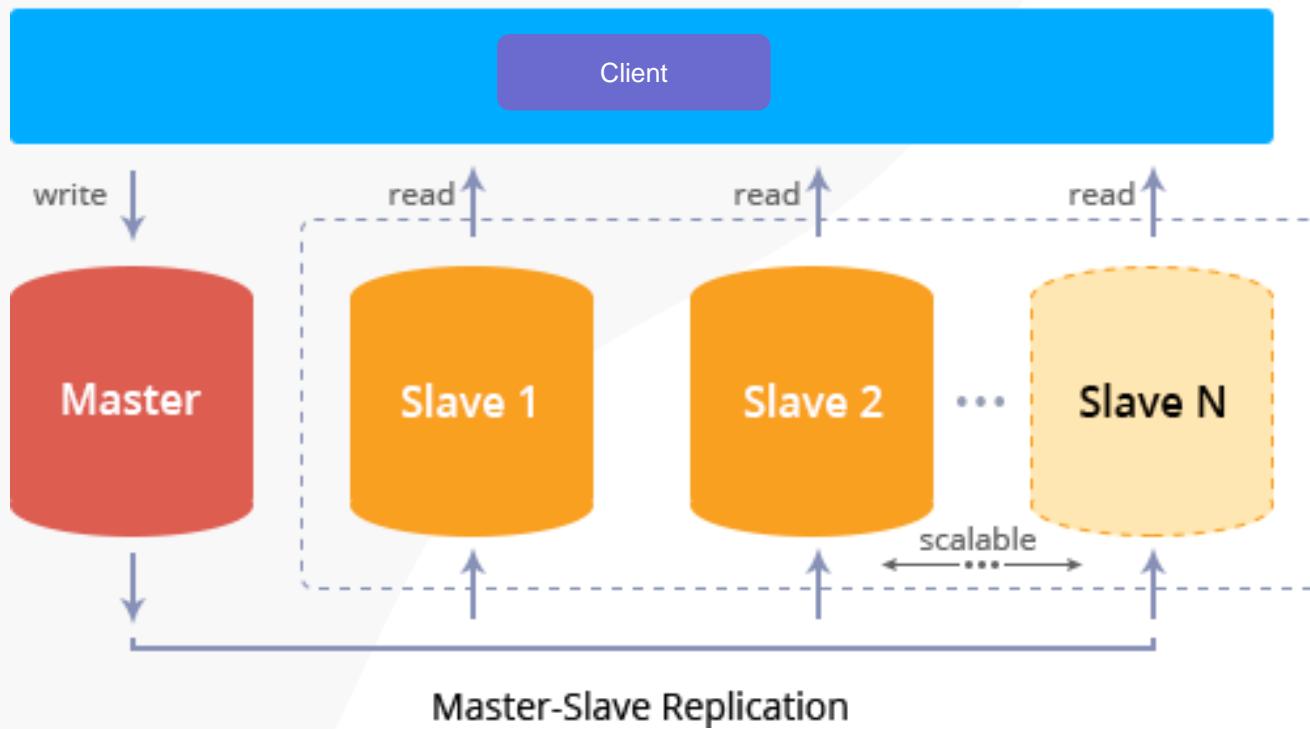
# Replicação Hierárquica



- Também conhecida como **single-master** ou **master-slave**;
- Topologia com um nó principal (replica primária / master) e N nós secundários (slaves);
- Operações de escrita (insert / update / delete) somente no nó primário;
- Replicação do log com as transações: síncrona ou assíncrona;
- Operação de leitura na(s) réplica(s) secundária(s): depende do SGBDD.

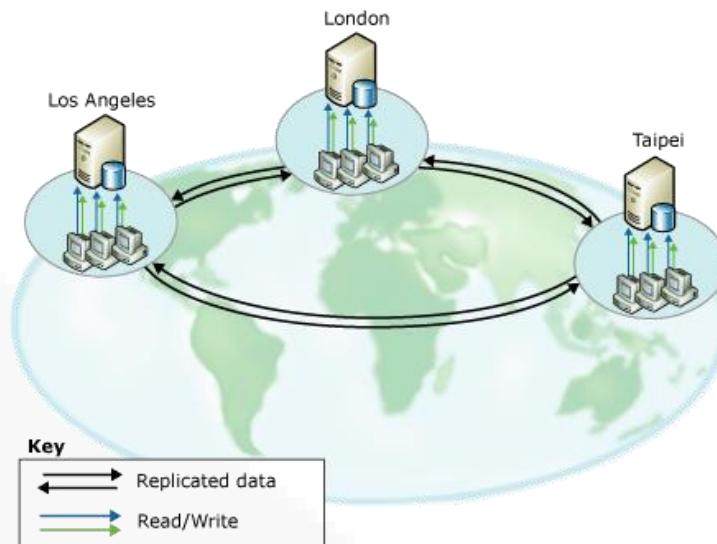
# Replicação Hierárquica

IGTI



# Replicação Peer-To-Peer (P2P)

- Também conhecida como **multi-master ou update-anywhere**
  - ✓ Topologia circular, cruzada, etc.
- Operações de escrita (insert / update / delete) em todos os nós.

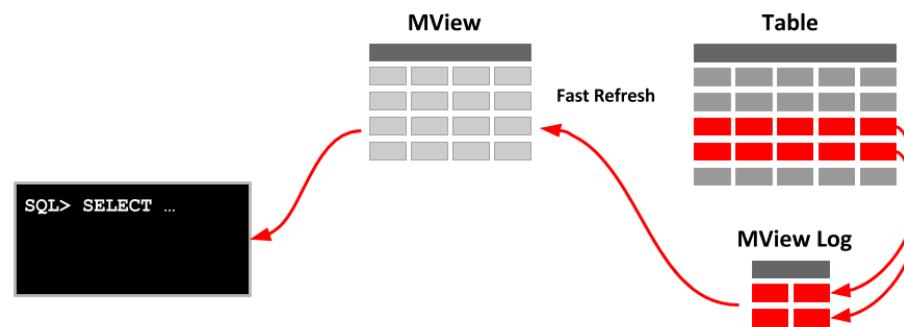


# Opções para Replicação de Dados

IGTI

## ▪ Oracle Materialized View

- Read only / Updatable\* (requer o MVLog);
- Dblink para outros SGBDs.



Tipo? Hierárquico  
Peer-to-Peer\*

- Assíncrono
- Síncrono

# Opções para Replicação de Dados

IGTI

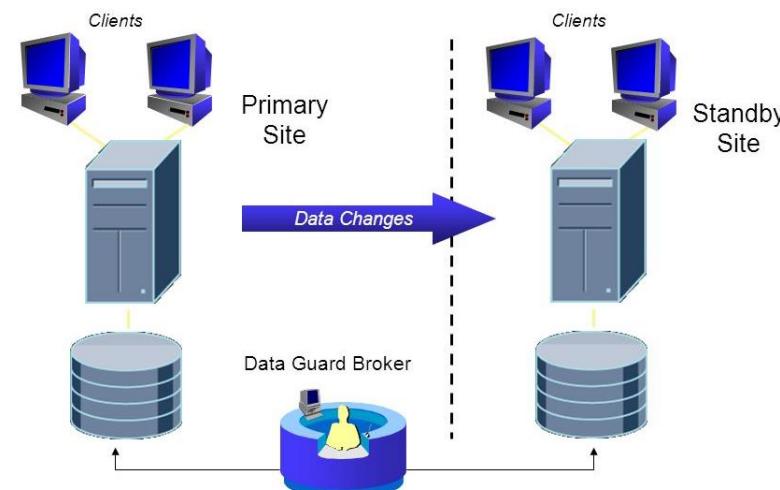
## ▪ Oracle Data Guard

- Replicação física dos dados;
- Escrita em 1 nó apenas;
- Standby em recovery.

Tipo?

Hierárquico

- Assíncrono
- Síncrono

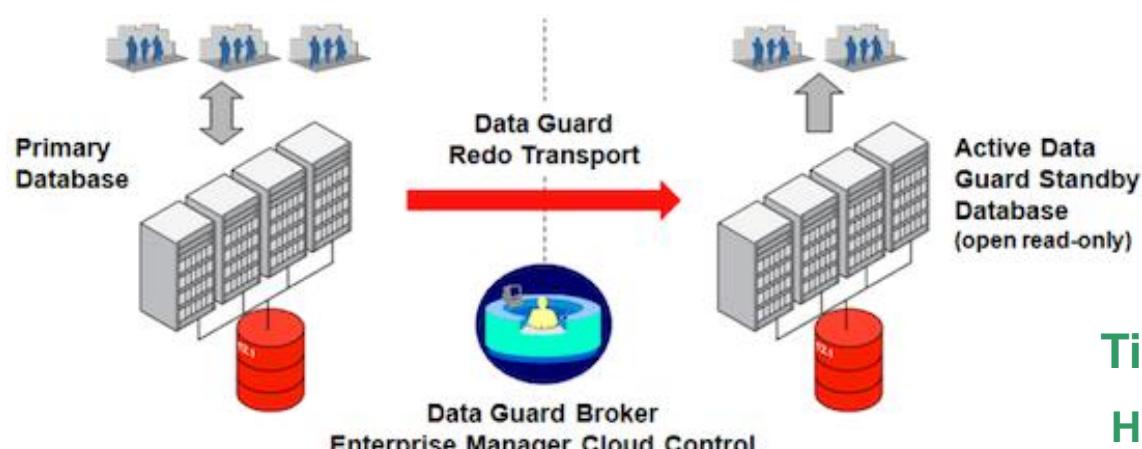


# Opções para Replicação de Dados

IGTI

## ▪ Oracle Active Data Guard

- Escrita apenas no servidor primário;
- Replicação das transações;
- Servidor secundário read-only.



Tipo?

Hierárquico

- Assíncrono
- Síncrono



# Opções para Replicação de Dados

IGTI

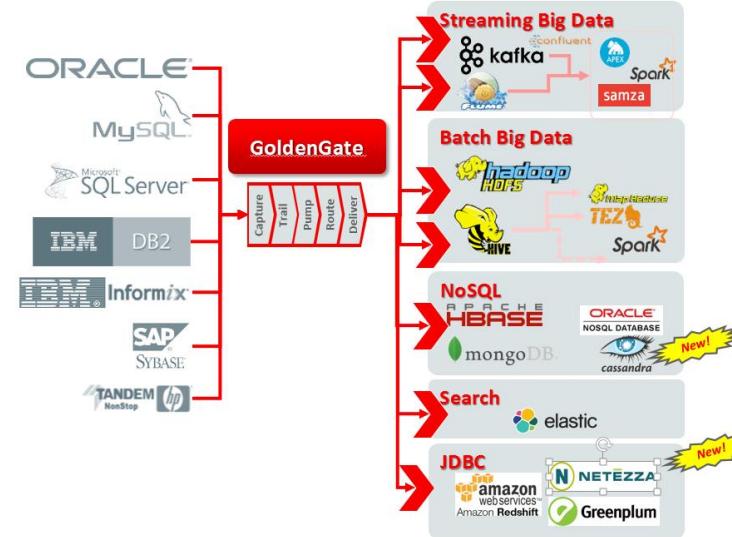
## ▪ Oracle Golden Gate

- Replica / Distribui:
  - Banco inteiro / tabela inteira / linhas / colunas.
- Diversas fontes e origem de dados:
  - Replicação / Distribuição lógica;
  - Backup de negócio;
  - Homogêneo: bidirecional.

Tipo?

Hierárquico  
Peer-to-Peer

- Assíncrono
- Síncrono

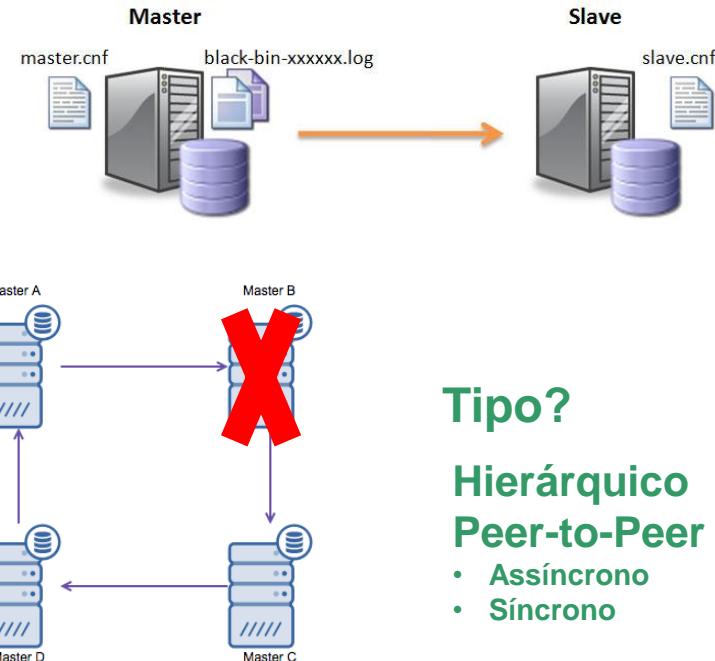


# Opções para Replicação de Dados

IGTI

## ▪ MySQL Replication

- Replicação do binary log;
- **Master-slave:**
  - Escrita apenas no servidor primário;
  - Servidor secundário read-only.
- **Master-master:**
  - Escrita em todos os nós;
  - Arquitetura anel
    - Um servidor só pode ser master de um slave e slave de um master;
    - Indisponibilidade nos pontos do anel(!).



Tipo?

Hierárquico  
Peer-to-Peer  
• Assíncrono  
• Síncrono



# Próxima Aula



- Tipos de Particionamento de Dados.

# Bancos de Dados Distribuídos

---

CAPÍTULO 1. AULA 1.5. TIPOS DE PARTICIONAMENTO DE DADOS

PROF. GUSTAVO AGUILAR

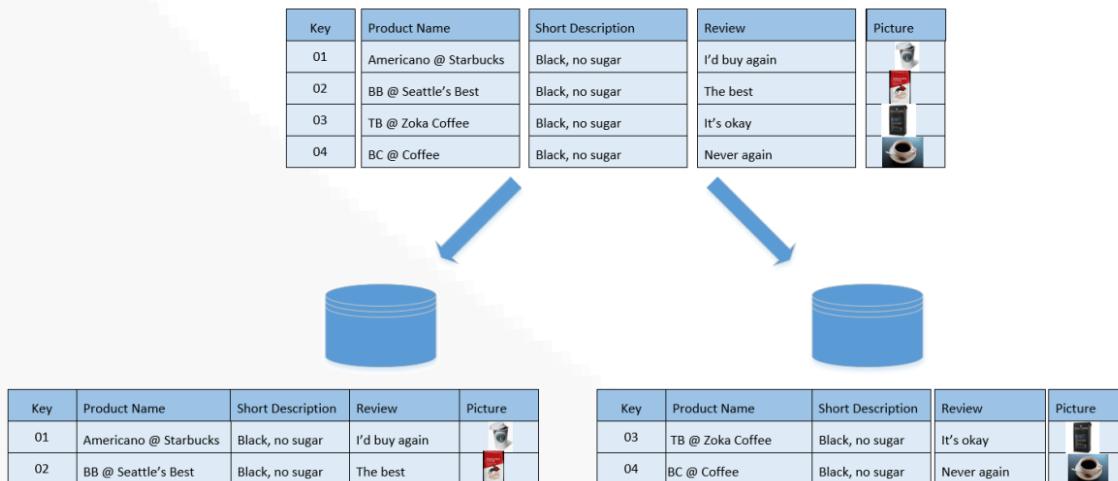
# Nesta Aula



- Particionamento Horizontal.
- Particionamento Vertical.
- Particionamento Misto.
- Opções para Particionamento de Dados.

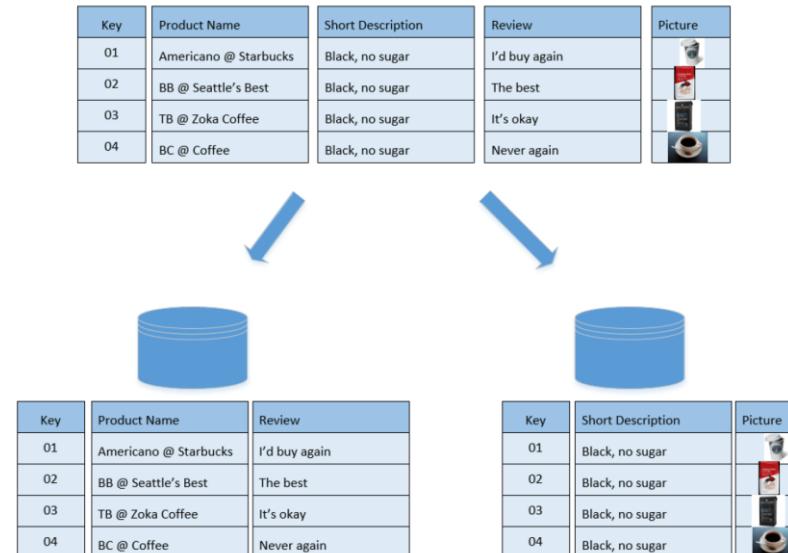
# Particionamento Horizontal

- Cada fragmento (shard) consiste em um subconjunto de linhas (tuplas);
- É definido pela operação de restrição (filtro):
  - ✓ Hash (hash key);
  - ✓ Faixa de valores.
- Muito utilizado em conjunto com a distribuição geográfica dos dados.



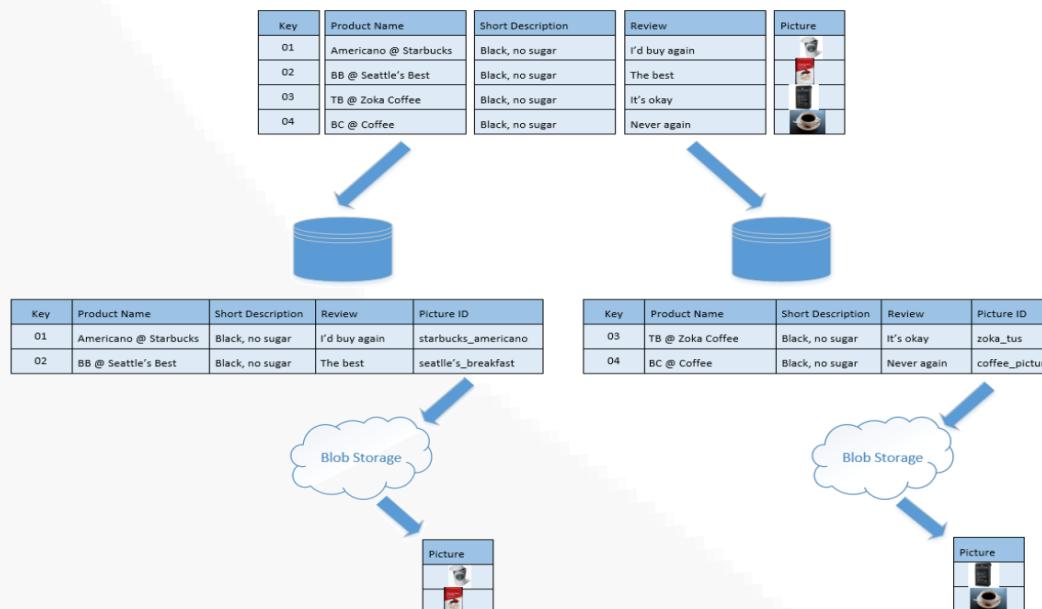
# Particionamento Vertical

- Cada fragmento (shard) consiste em um subconjunto de atributos (colunas);
- É definido pela operação de projeção (select):
  - Afinidade dos atributos;
  - Confidencialidade dos dados;
  - Armazenamento;
  - Localização.
- Atributo(s) da chave primária e atributos obrigatórios devem ser replicados em todos shards.



# Particionamento Misto

- Utilização das técnicas de particionamento horizontal e vertical juntas;
- É definido pela operação de restrição e pela de projeção;
- Muito comum de ser usado em grandes conjuntos de dados com diferentes tipos de dados, como por exemplo, CLOB, BLOB e XML.



# Opções para Particionamento

- **MySQL Cluster (Carrier Grade Edition - CGE)**

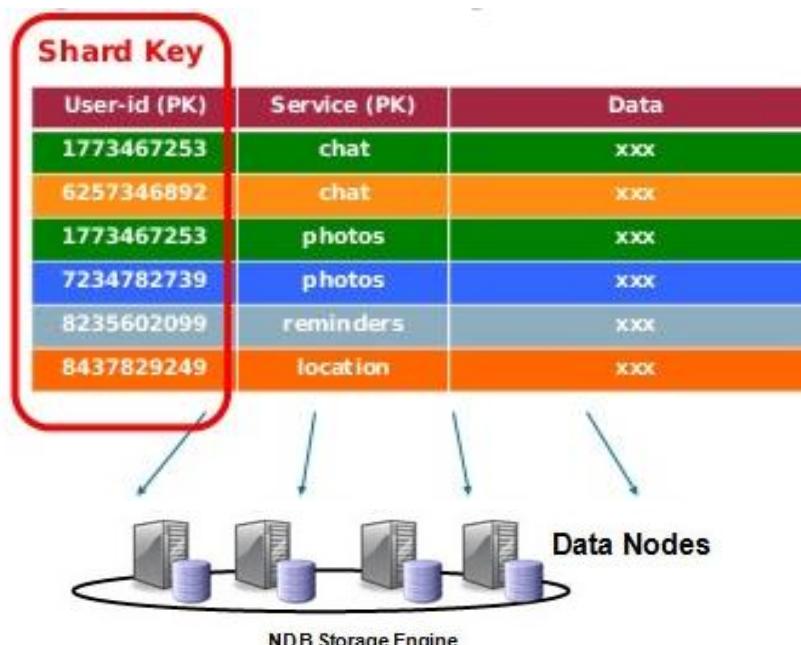
- Leitura e escrita em todos nós;
- Escalabilidade: horizontal e vertical;
- NDB Cluster;
- Usa arquitetura de sharding;
- Alta disponibilidade:
  - Replicação dos shards.

Tipo?

Peer-to-Peer

- Assíncrono

Horizontal



# Conclusão



- **Escalar horizontalmente o workload de leitura:**
  - Replicação hierárquica.
- **Escalar horizontalmente o workload de leitura e escrita:**
  - Replicação peer-to-peer.
- **Escalar verticalmente:**
  - Todos tipos de replicação.
- **Ambiente escalável com consistência eventual:**
  - Replicação assíncrona.
- **Ambiente escalável com consistência total:**
  - Replicação síncrona.

# Próxima Aula



- Teorema de CAP e Propriedades BASE.

# Bancos de Dados Distribuídos

---

CAPÍTULO 1. AULA 1.6. TEOREMA DE CAP E PROPRIEDADES BASE

PROF. GUSTAVO AGUILAR

# Nesta Aula



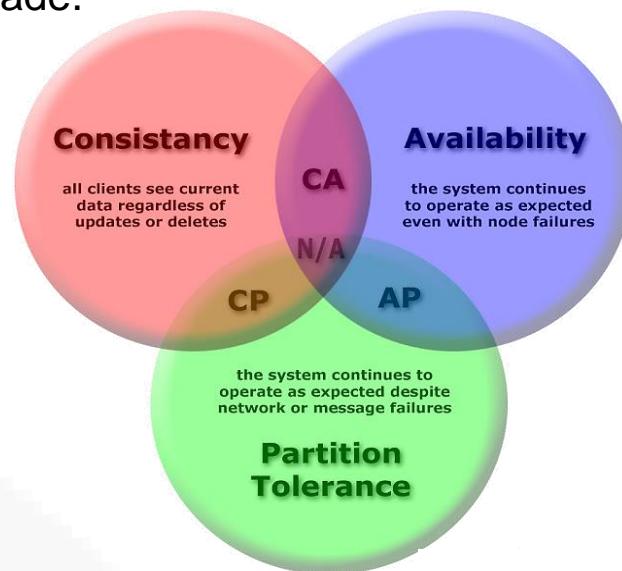
- Teorema de CAP.
- Propriedades BASE.

# Teorema de CAP

- Criado por Eric Brewer em 2000;
- **Três principais requisitos sistêmicos** em um ambiente distribuído:
  - Consistency, Availability, Partition tolerance;
  - **Consistência**: todos veem os mesmos dados;
  - **Disponibilidade**: o sistema está disponível quando solicitado, independentemente de ter ocorrido uma falha em alguns dos servidores;
  - **Tolerância à Partição**: perda de conexão entre os nós ou perda de dados em um determinado nó não causam impacto.
- Sistema distribuído pode garantir **apenas dois desses requisitos**.

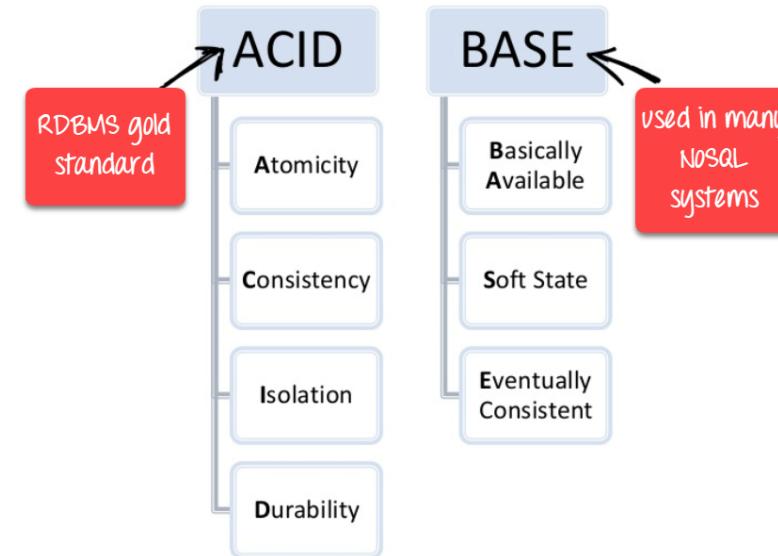
# Teorema de CAP

- Nenhum sistema distribuído está protegido contra falhas de rede:
  - Portanto, a partição (falha) geralmente deve ser tolerada.
- Podendo haver falhas (partições), são dadas duas opções:
  - Tolerância a Falhas + Consistência;
  - Tolerância a Falhas + Disponibilidade.



# Propriedades BASE

- Modelo alternativo de consistência
  - **Basically Available**: haverá uma resposta a qualquer solicitação, mesmo que de falha;
  - **Soft state**: o estado do sistema pode mudar ao longo do tempo, mesmo em momentos sem entrada de dados → necessário para a consistência eventual;
  - **Eventual consistency**: dados propagados para todos os nós, assincronamente.



“*BASE: An Acid Alternative*” - Dan Pritchett

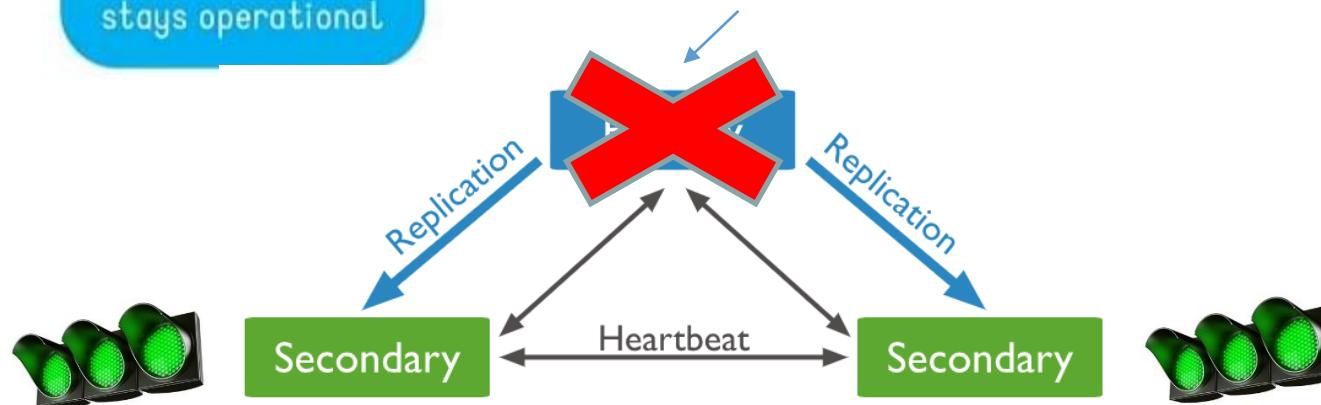
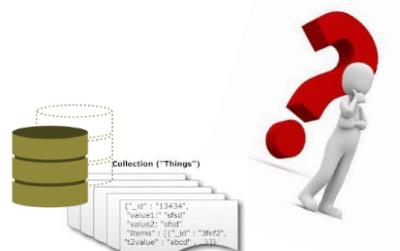
<https://queue.acm.org/detail.cfm?id=1394128>

# Propriedades BASE

**BASE**

= Basically Available, Soft state, Eventual consistency

If a node fails,  
part of the data  
will not be  
available, but the  
entire data layer  
stays operational



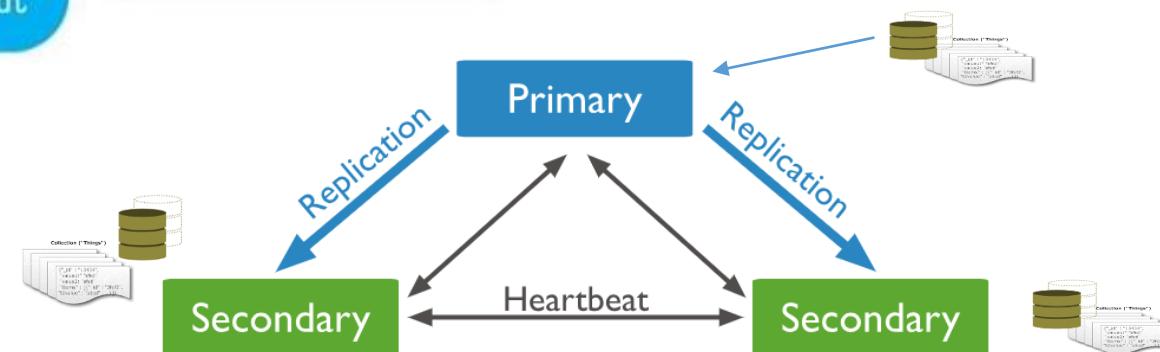
# Propriedades BASE

**BASE** = Basically Available, Soft state, Eventual consistency

If a node fails, part of the data will not be available, but the entire data layer stays operational

The state of the system may change over time, even without input

The system becomes consistent at some later time



# Próxima Aula



- ❑ Capítulo 2 - Distribuição de Dados no SQL Server.

# Performance e Otimização

Capítulo 2. Distribuição de Dados no SQL Server

PROF. GUSTAVO AGUILAR

# Distribuição de Dados no SQL Server

---

CAPÍTULO 2. AULA 2.1. PARTITIONED VIEW

PROF. GUSTAVO AGUILAR

# Nesta Aula



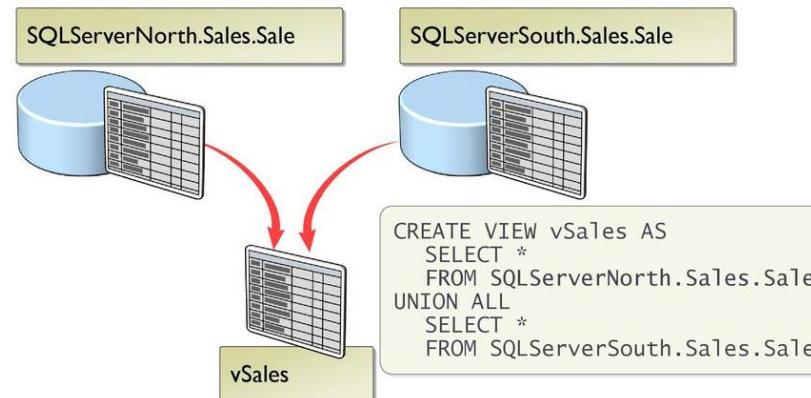
- Partitioned View.
- Demonstração.

# Partitioned View

- Homogêneo ou Heterogêneo → acessa outros SGBDs via linked server;
  - ✓ Sistema de Bancos de Dados Federados.
- Updatable (atualizável) → algumas restrições.
- Filtro por linhas e/ou colunas;
- Podem ser indexadas (!).

Tipo?

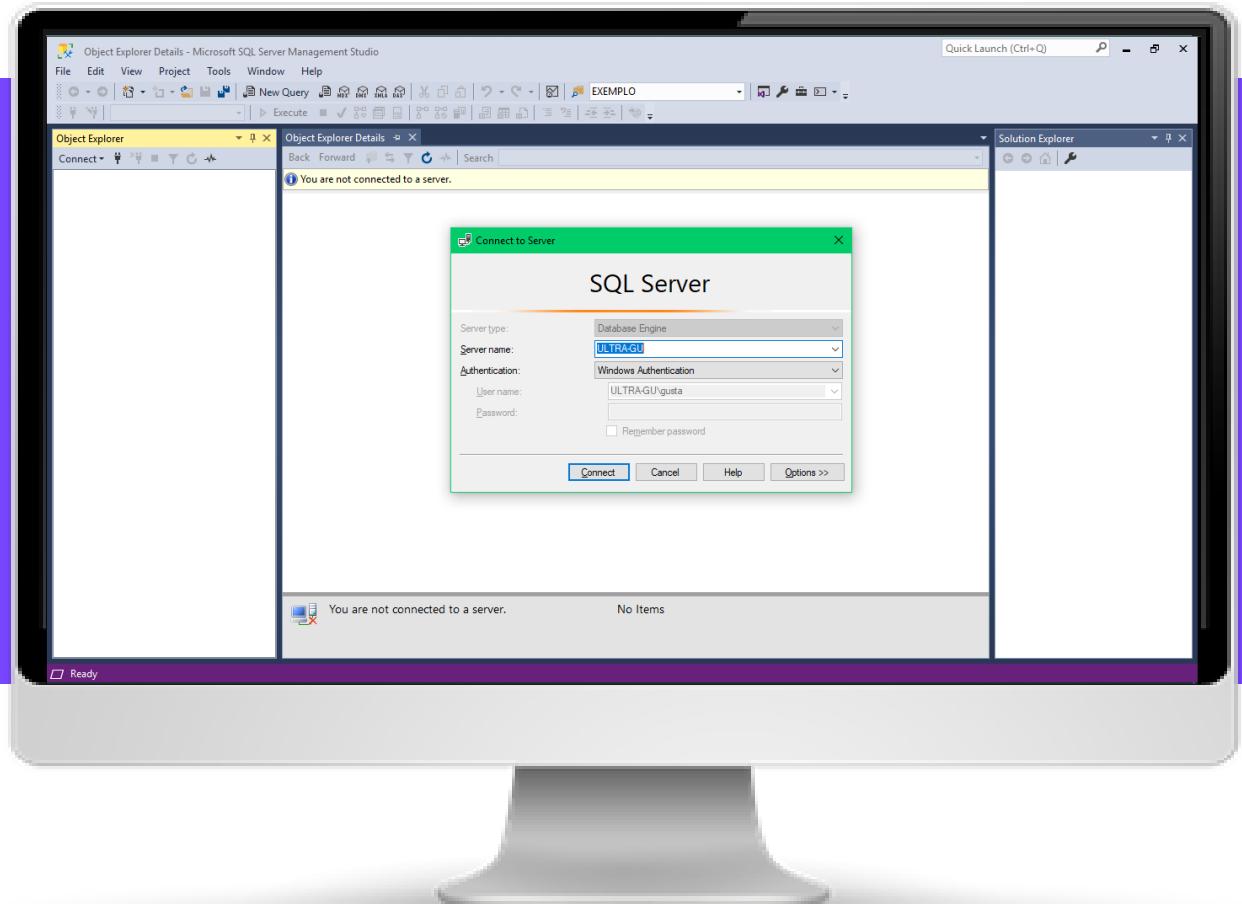
Horizontal  
Vertical



# Partitioned View



 Demo

A large white play button icon is positioned to the left of the word "Demo". The background behind the text is a solid purple color.

# Próxima Aula



- ❑ Log Shipping e SQL Server Replication.

# Distribuição de Dados no SQL Server

---

CAPÍTULO 2. AULA 2.2. LOG SHIPPING E SQL SERVER REPLICATION

PROF. GUSTAVO AGUILAR

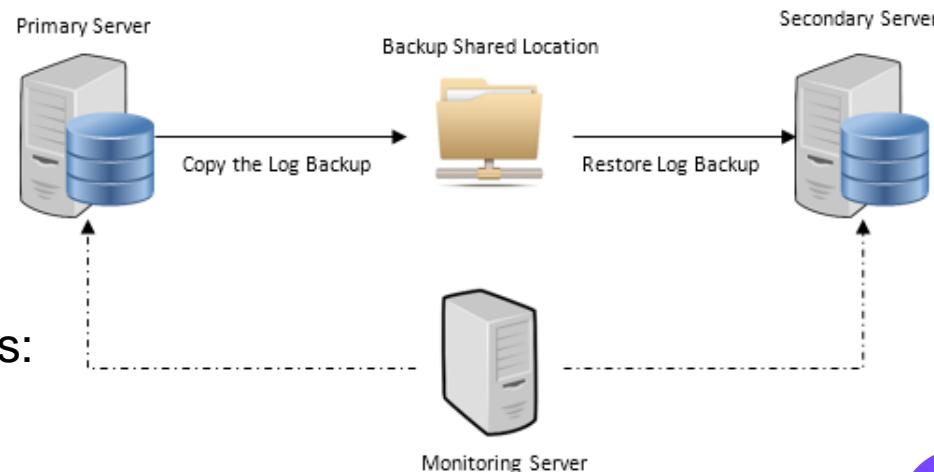
# Nesta Aula



- Log Shipping.
- SQL Server Replication.
- Demonstração.

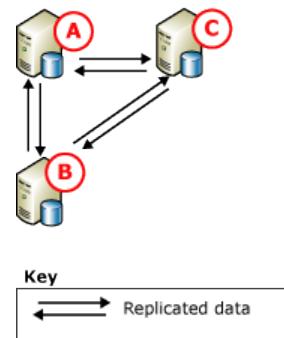
# Log Shipping

- Escrita apenas no servidor primário;
- Assíncrono;
- Backup / restore de logs:
  - Requer local compartilhado.
- Servidor secundário em 2 modos:
  - **No recovery mode**: sem leituras;
  - **Standby mode**: permite leituras.
- Ideal é que as versões do SQL sejam iguais:
  - Versão do secundário pode ser superior, mas em caso de failover, não tem fallback.

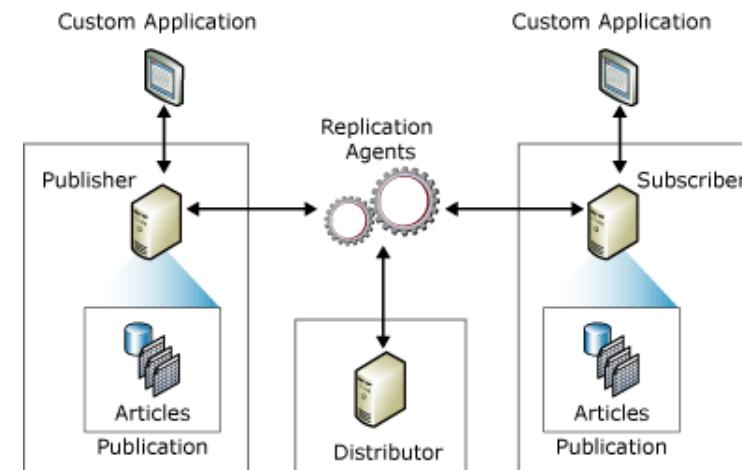


# SQL Server Replication

- Replica / Distribui: banco inteiro / tabela inteira / linhas / colunas;
- Snapshot, Transacional;
- Merge: escrita em qualquer nó;
- Publisher: SQL e Oracle;
- Subscriber: SQL, Oracle e DB2.



Tipo?



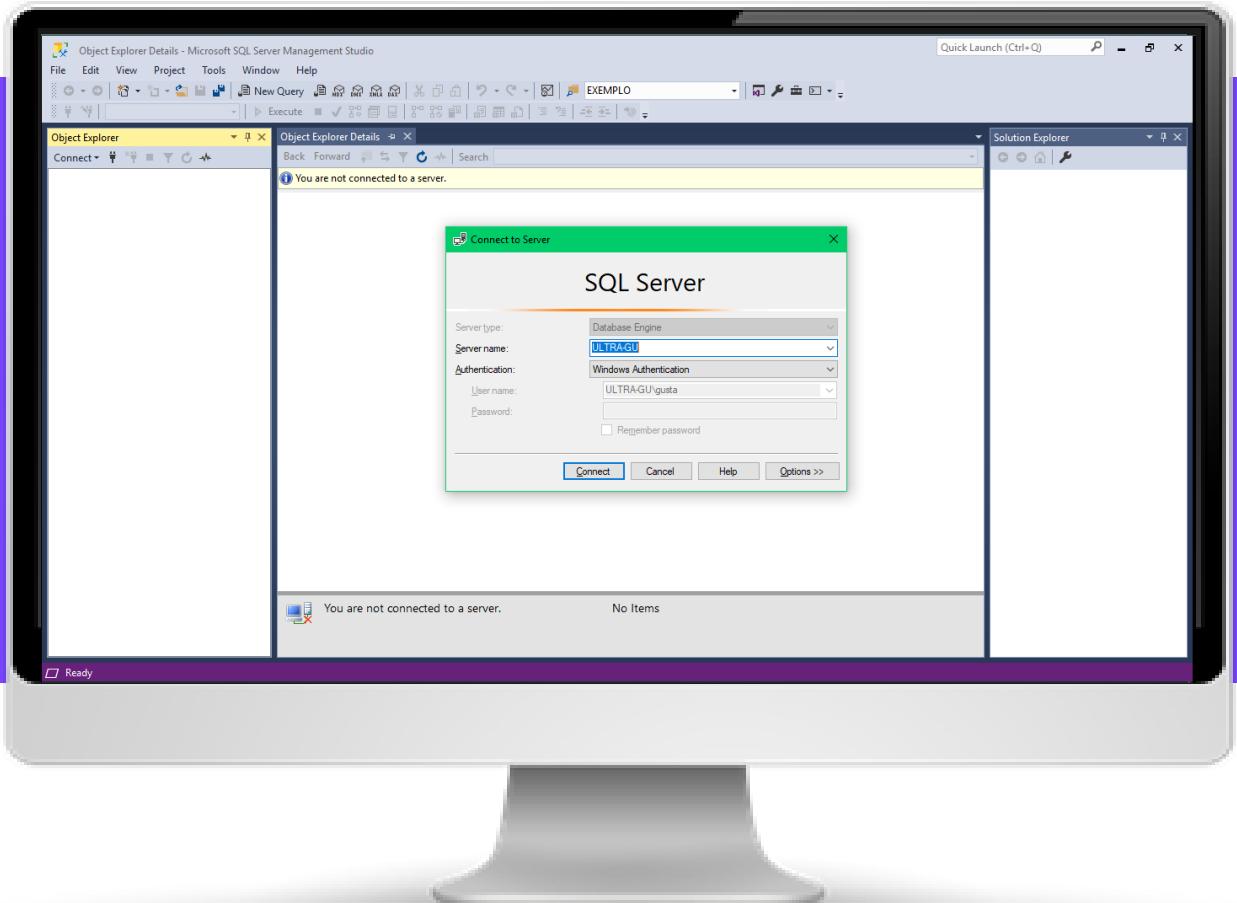
Hierárquico / Peer-to-Peer  
• Assíncrono  
Horizontal / Vertical



# Log Shipping e SQL Server Replication



 Demo



# Próxima Aula



- ❑ Database Mirroring e AlwaysOn Availability Groups.

# Distribuição de Dados no SQL Server

---

CAPÍTULO 2. AULA 2.3. DATABASE MIRRORING E ALWAYSON AVAILABILITY GROUPS

PROF. GUSTAVO AGUILAR

# Nesta Aula



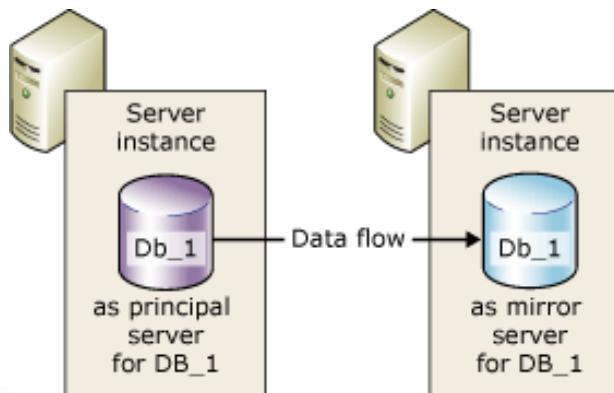
- Database Mirroring.
- AlwaysOn Availability Groups.

# Database Mirroring

- Replicação das transações existentes no transaction log;
- Não requer pasta compartilhada;
- Escrita apenas no servidor primário;
- Servidor secundário em recovery mode (não acessível para leituras);
- Replicação Assíncrona ou Síncrona;

## ✓ Assíncrona

- Mais performática;
- Apenas failover manual.



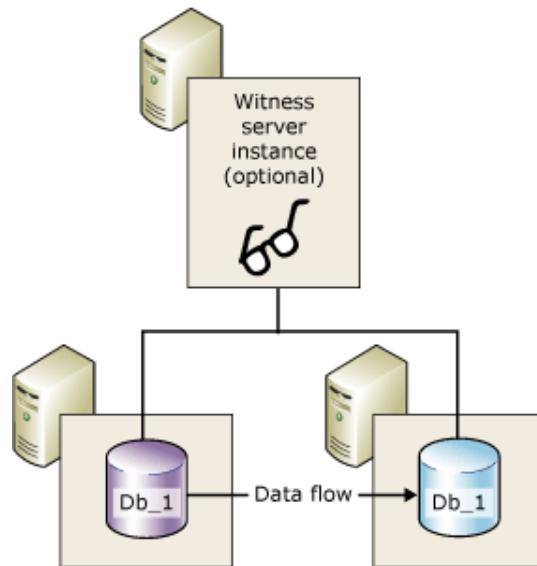
# Database Mirroring

- Replicação Assíncrona ou Síncrona;

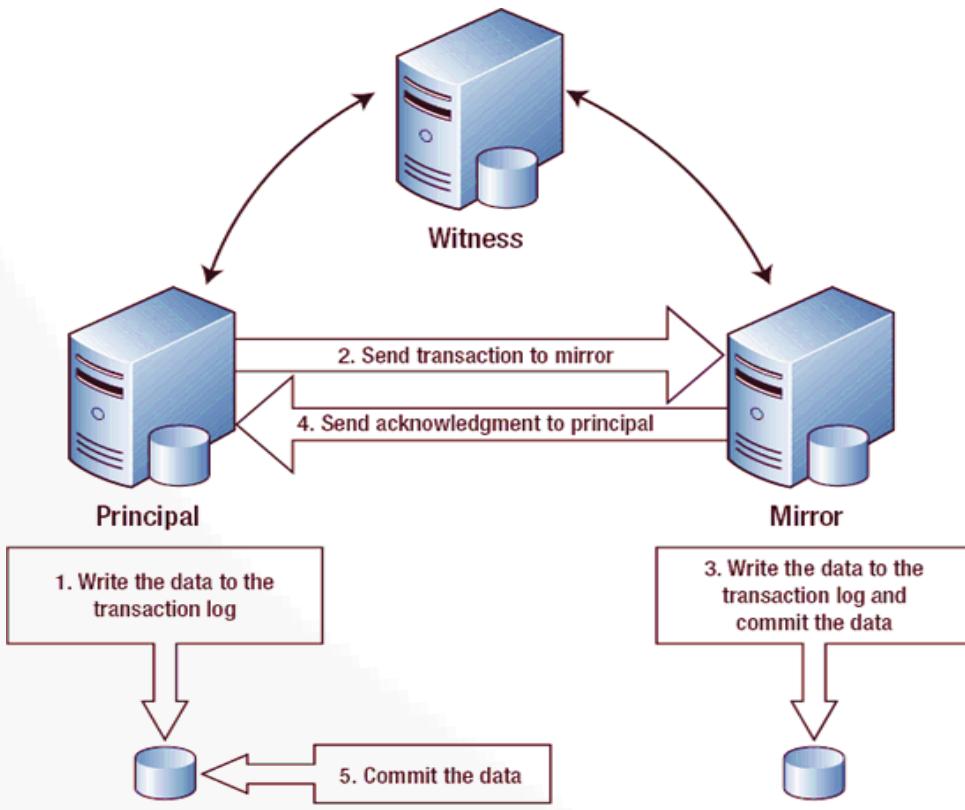
- ✓ **Síncrona**

- Mais segura;
  - Menos performática;
  - Failover manual;
  - Failover automático:

- ✓ Requer witness.



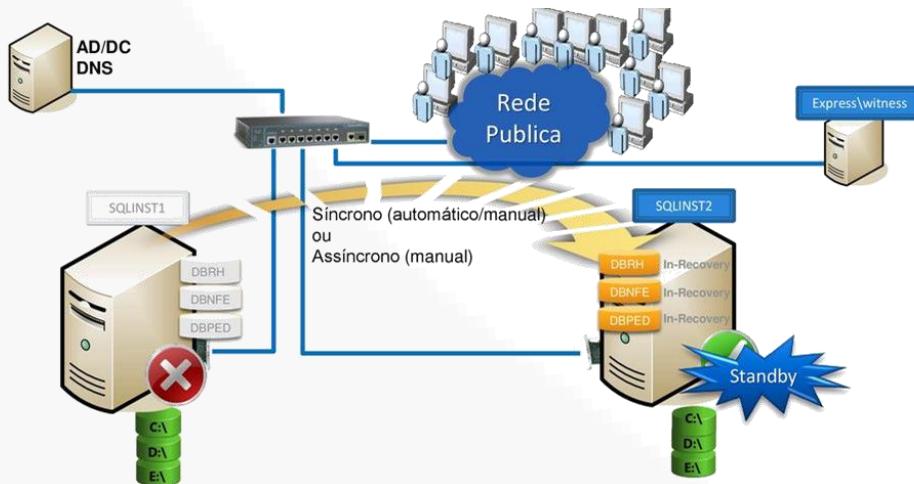
# Database Mirroring



\*Síncrono com failover automático

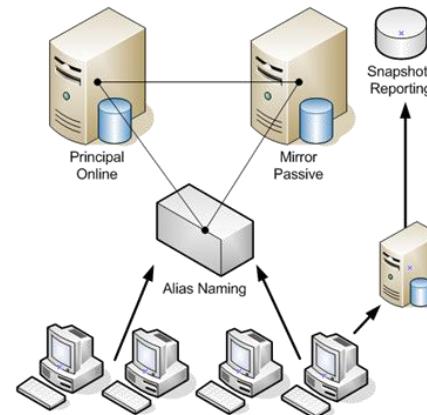
# Database Mirroring

- Muito usado para disaster recovery / alta disponibilidade / PCNs;
- Ideal é que as versões de SQL sejam iguais:
  - ✓ Versão do mirror pode ser maior, mas em caso de failover, não tem fallback.



Tipo?

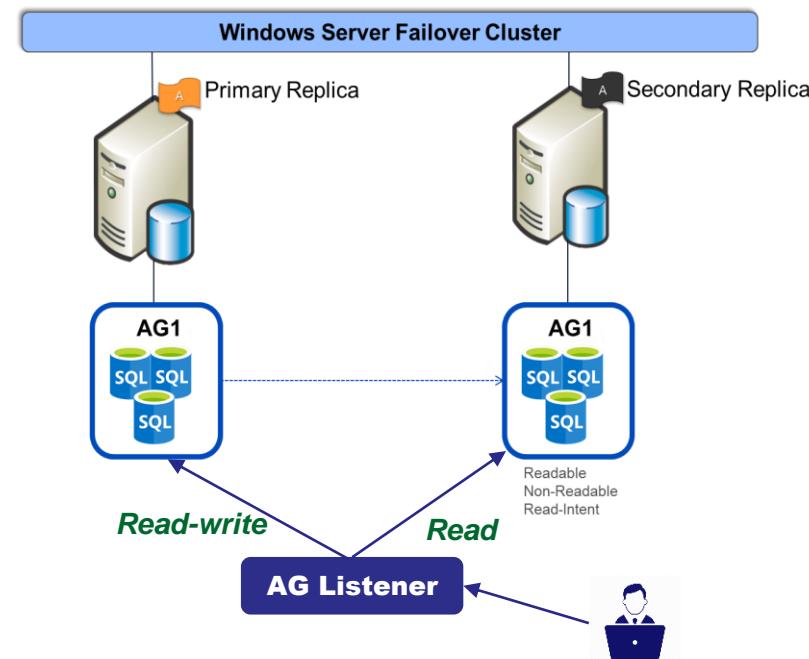
Hierárquico  
• Assíncrono  
• Síncrono



# AlwaysOn Availability Groups

- Replicação das transações:
  - Modo Síncrono → mais seguro;
  - Modo Assíncrono → mais performático.
- Escrita apenas no servidor primário;
- Servidor secundário permite leitura:
  - ✓ Backup no secundário.
- Listener: transparência de localização.

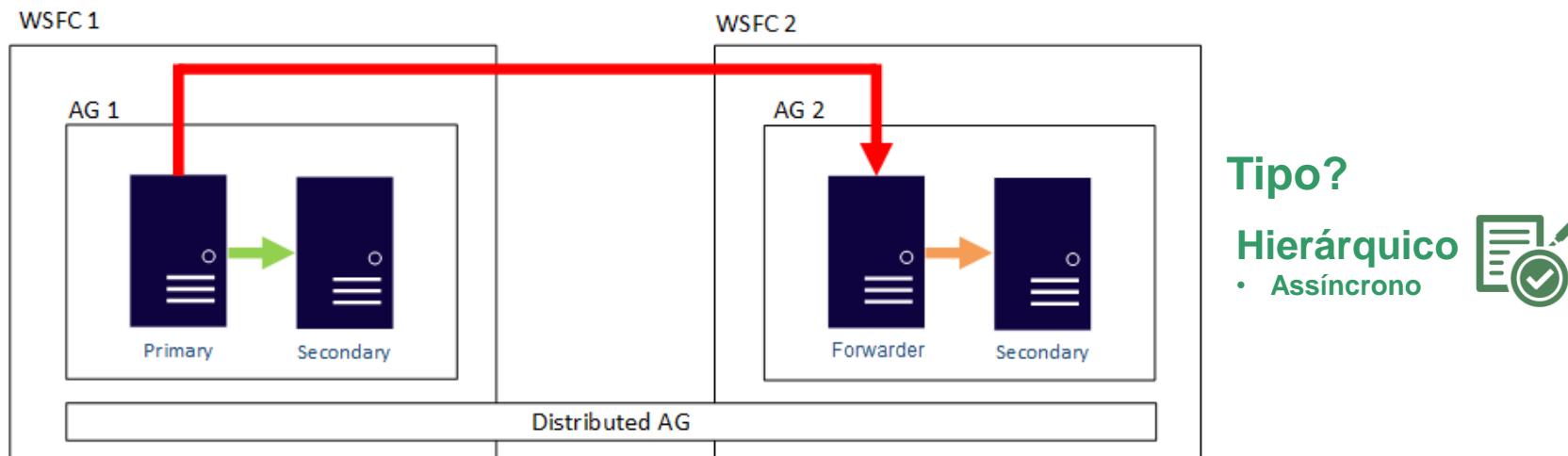
**Tipo? Hierárquico**  
• Assíncrono  
• Síncrono



# AlwaysOn Availability Groups

IGTI

- **Distributed Availability Groups (DAG):** a partir do SQL 2016;
- Escrita somente na réplica primária do AG primário;
- Listener não é único e não há failover automático dos AGs entre sites.



**Tipo?**

**Hierárquico**  
• Assíncrono



# Próxima Aula



- ❑ Capítulo 3 - Banco de Dados Distribuído como Serviço.

# Performance e Otimização

Capítulo 3. Banco de Dados Distribuído como Serviço

PROF. GUSTAVO AGUILAR

# Banco de Dados Distribuído como Serviço

---

CAPÍTULO 3. AULA 3.1. AZURE SQL DATABASE MANAGED INSTANCE

PROF. GUSTAVO AGUILAR

# Nesta Aula



- ❑ Ofertas de Instâncias Gerenciada.
- ❑ Recursos da Instância Gerenciada.
- ❑ Criando uma SQL Managed Instance.
- ❑ Usando uma SQL Managed Instance.

# Ofertas de Instância Gerenciada



- **General Purpose**

- **Business Critical**

- Maior desempenho e disponibilidade;
- Suporta recursos In-Memory OLTP do SQL;
- Leitura em réplicas secundárias;
- Mais memória RAM por vCore;
- Usa storage atachado diretamente (sem NAS)
  - Menor latência nas operações de I/O.
- Uso das licenças de SQL já compradas para reduzir o custo.

# Recursos da Instância Gerenciada

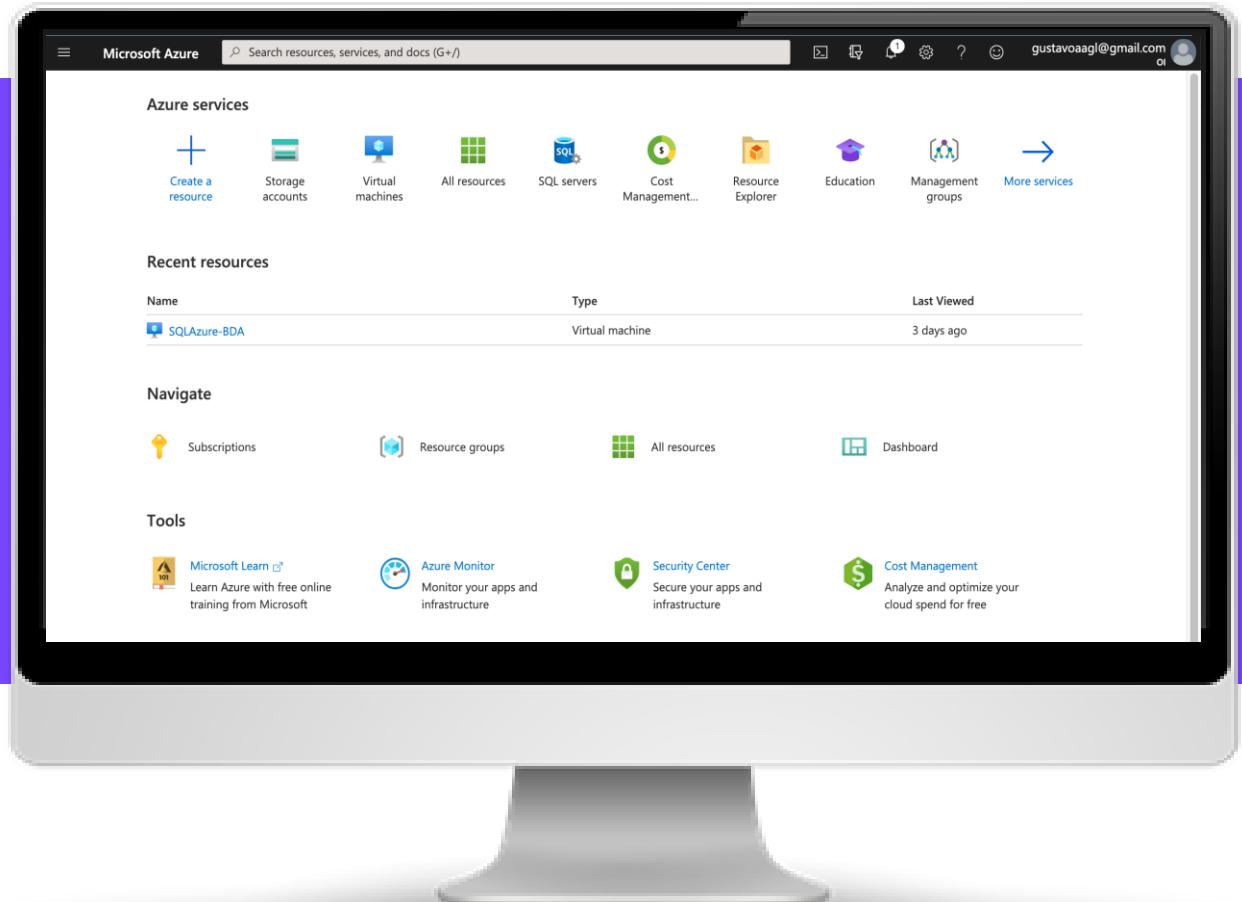


- 99.99% de disponibilidade;
- Atualizações automáticas de patches Windows / SQL;
- Backup gerenciado pelo Azure
  - Possível gerar um backup manual “copy only” para o Azure Storage.
- Automatic tuning:
  - Identificar queries onerosas;
  - Forçar o último plano execução bom;
  - Adicionar índices;
  - Remover índices.

# Criando e Usando uma SQL Managed Instance



 Demo



# Próxima Aula



- Azure SQL Database.

# Banco de Dados Distribuído como Serviço

---

CAPÍTULO 3. AULA 3.2. AZURE SQL DATABASE

PROF. GUSTAVO AGUILAR

# Nesta Aula



- ❑ Azure SQL Database.
- ❑ Criando um Azure SQL Database.
- ❑ Usando um Azure SQL Database.

# Azure SQL Database

- Possui 2 modelos de implantação;
- **Single Database**
  - Banco de dados isolado;
  - Totalmente gerenciado pelo Azure.
- **Elastic Pool**
  - Coleção de single databases;
  - Com um conjunto compartilhado de recursos (CPU / RAM).

# Azure SQL Database

- Modelos de contratação
  - **Baseado em vCore:** permite selecionar a quantidade de CPU, memória RAM e velocidade do storage.
  - **Baseado em DTU (Database Transaction Units):** combinação de recursos de computação, memória e I/O em três camadas de serviço, para suportar cargas de trabalho de banco de dados leves a pesadas.
  - **Serverless:**
    - Dimensiona automaticamente os recursos com base na demanda da carga de trabalho e cobra pela quantidade de recursos usados por segundo.
    - Pausa automaticamente os bancos de dados durante os períodos inativos → cobra o storage.

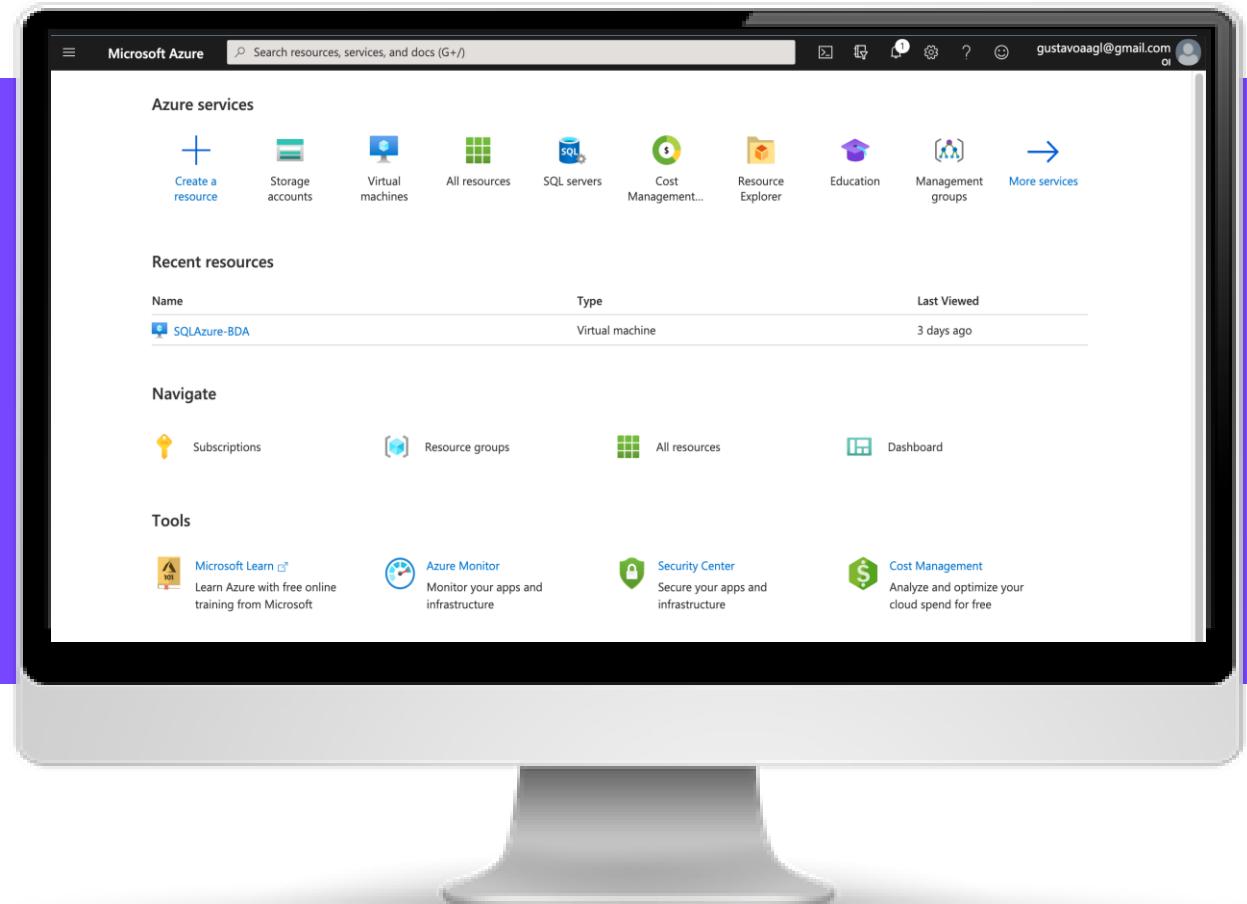
# Azure SQL Database

- Disponível em 3 camadas de serviço (service tiers);
- **General Purpose / Standard:** para cargas de trabalho comuns.
- **Business Critical / Premium:**
  - Para aplicações OLTP com alta taxa de transações;
  - Cenários que necessitam de baixa latência de I/O;
  - Oferece a maior resiliência a falhas usando várias réplicas isoladas.
- **Hyperscale:**
  - Para bancos de dados OLTP muito grandes;
  - Cenários que necessitem de dimensionamento automático de armazenamento.

# Criando e Usando um Azure SQL Database



 Demo



# Próxima Aula



- ❑ Azure Cosmos DB.

# Banco de Dados Distribuído como Serviço

---

CAPÍTULO 3. AULA 3.3. AZURE COSMOS DB

PROF. GUSTAVO AGUILAR

# Nesta Aula



- ❑ Pré-requisitos para Criação.
- ❑ Criando um Azure Cosmos DB (SQL).
- ❑ Usando um Azure Cosmos DB (SQL).

# Pré-requisitos para Criação

## ▪ Azure Cosmos DB Account

- Recurso que atua como uma entidade organizacional;
- Cada conta está associada a um dos vários modelos de dados aos quais o Azure Cosmos DB oferece suporte;
- Quantas contas precisar.



# Pré-requisitos para Criação

## ▪ Request Unit (RU)

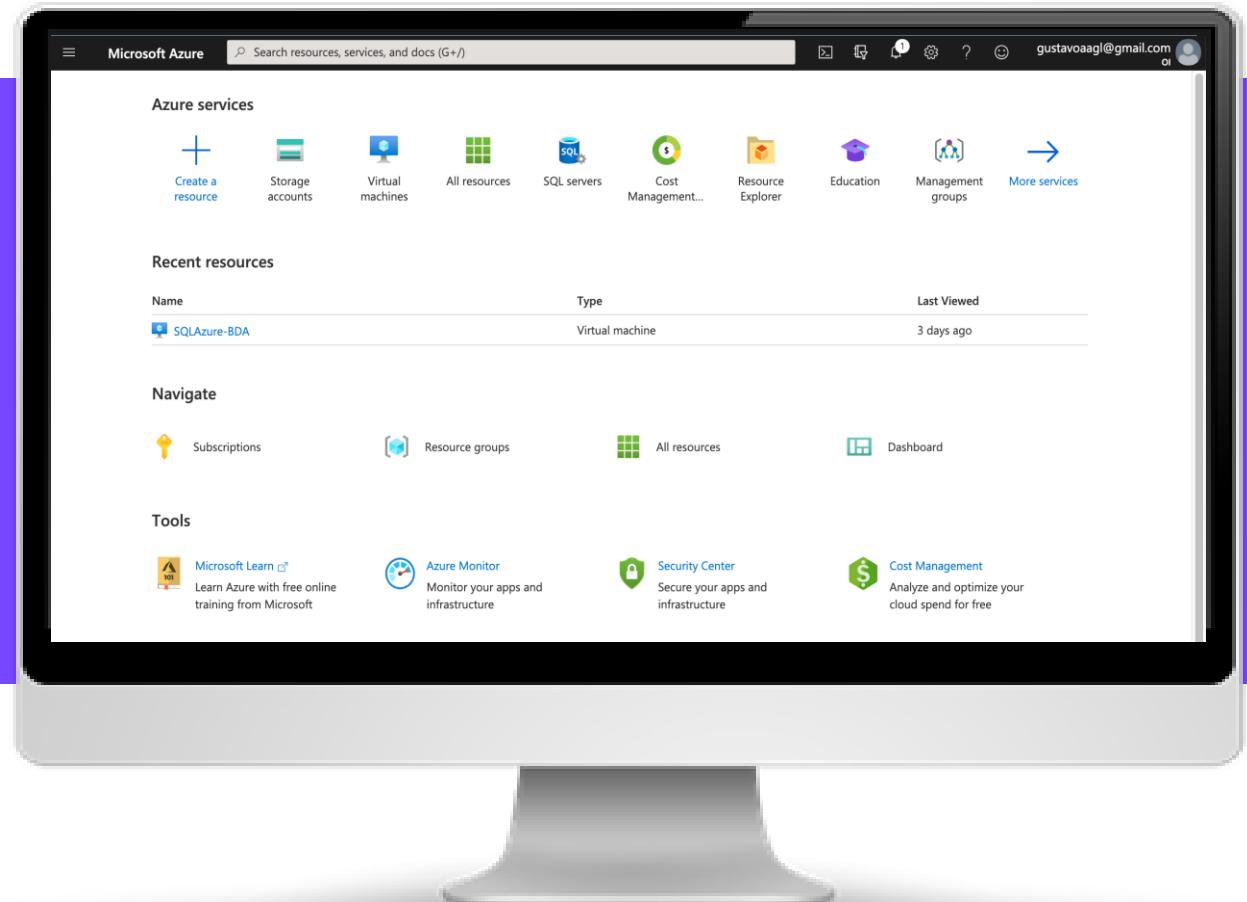
- Medida da taxa de transferência (throughput) por segundo;
- Reservar o número de RU/s que deseja que o Azure Cosmos DB provisione com antecedência, para que ele possa lidar com a carga estimada e você possa aumentar ou diminuir sua RU/s a qualquer .

Item size	Reads/second	Writes/second	Request units
1 KB	500	100	$(500 * 1) + (100 * 5) =$ 1,000 RU/s
1 KB	500	500	$(500 * 1) + (500 * 5) =$ 3,000 RU/s
4 KB	500	100	$(500 * 1.3) + (100 * 7) =$ 1,350 RU/s
4 KB	500	500	$(500 * 1.3) + (500 * 7) =$ 4,150 RU/s
64 KB	500	100	$(500 * 10) + (100 * 48) =$ 9,800 RU/s
64 KB	500	500	$(500 * 10) + (500 * 48) =$ 29,000 RU/s

# Criando e Usando um Azure Cosmos DB (SQL)



 Demo



# Próxima Aula



- ❑ Bancos de Dados Open Source no Azure.

# Banco de Dados Distribuído como Serviço

---

CAPÍTULO 3. AULA 3.4. BANCOS DE DADOS OPEN SOURCE NO AZURE

PROF. GUSTAVO AGUILAR

# Nesta Aula

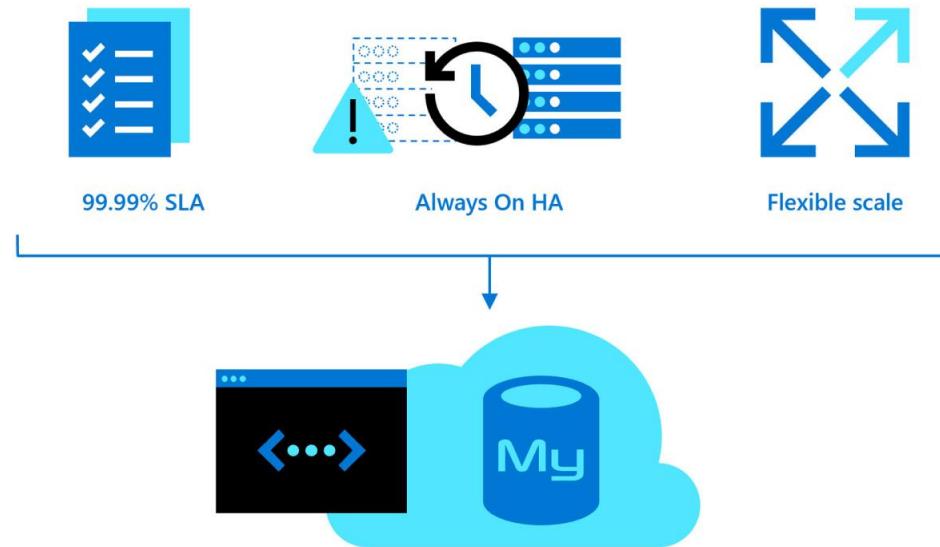


- ❑ Azure Database for MySQL.
- ❑ Azure Database for MariaDB.
- ❑ Azure Database for PostgreSQL.

# Bancos de Dados Open Source no Azure

IGTI

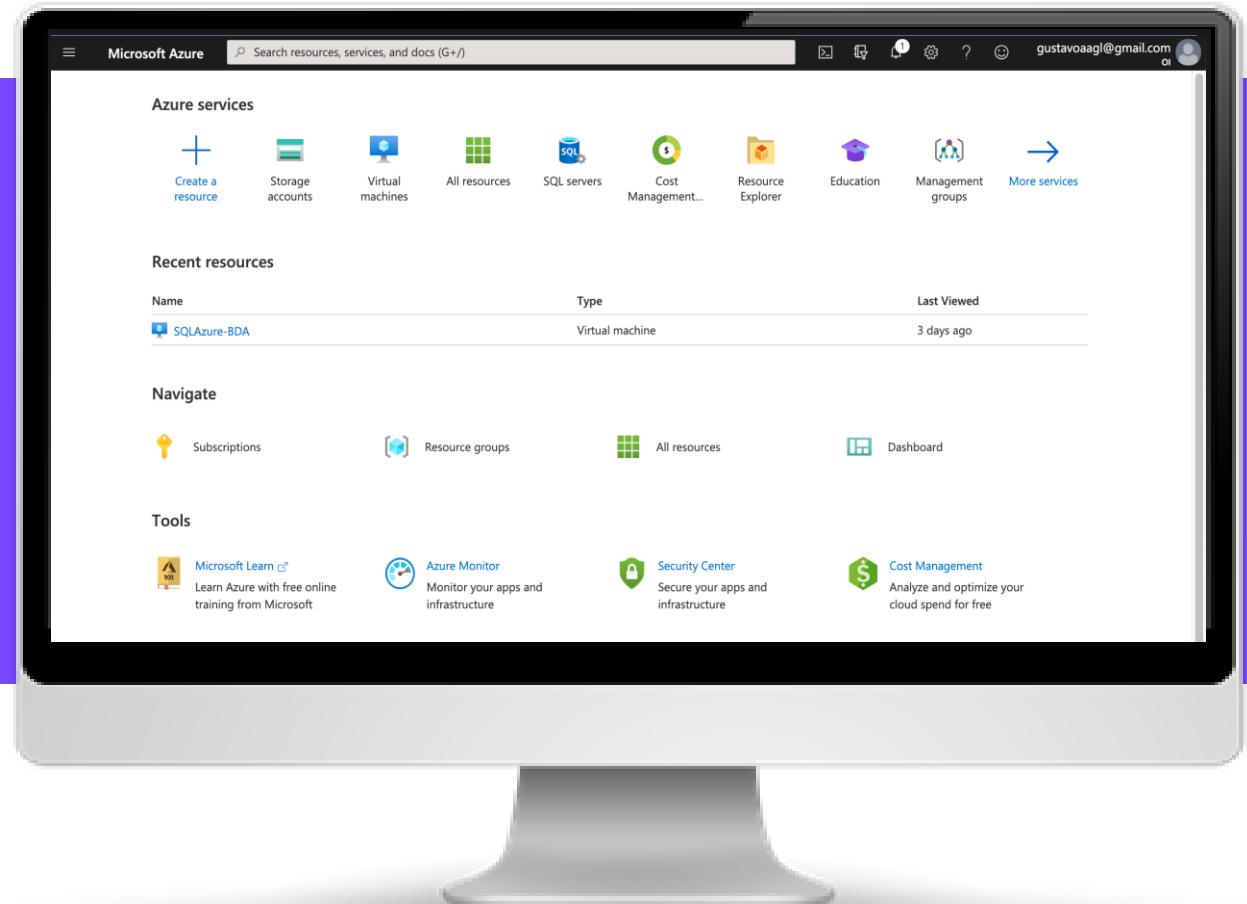
- Mecanismos para deploy:
  - Azure Portal;
  - PowerShell;
  - ARM templates;
  - Azure CLI.
- Azure Database for MySQL / MariaDB
  - Single Database / Replicação.
- Azure Database for PostgreSQL
  - Single Database / Hyperscale.



# Criando Azure Database for MySQL e for PostgreSQL



 Demo



# Próxima Aula



- ❑ Capítulo 4 - Distribuição de Dados no MongoDB.

# Performance e Otimização

Capítulo 4. Distribuição de Dados no MongoDB

PROF. GUSTAVO AGUILAR

# Distribuição de Dados no MongoDB

---

CAPÍTULO 4. AULA 4.1. REPLICA SET

PROF. GUSTAVO AGUILAR

# Nesta Aula



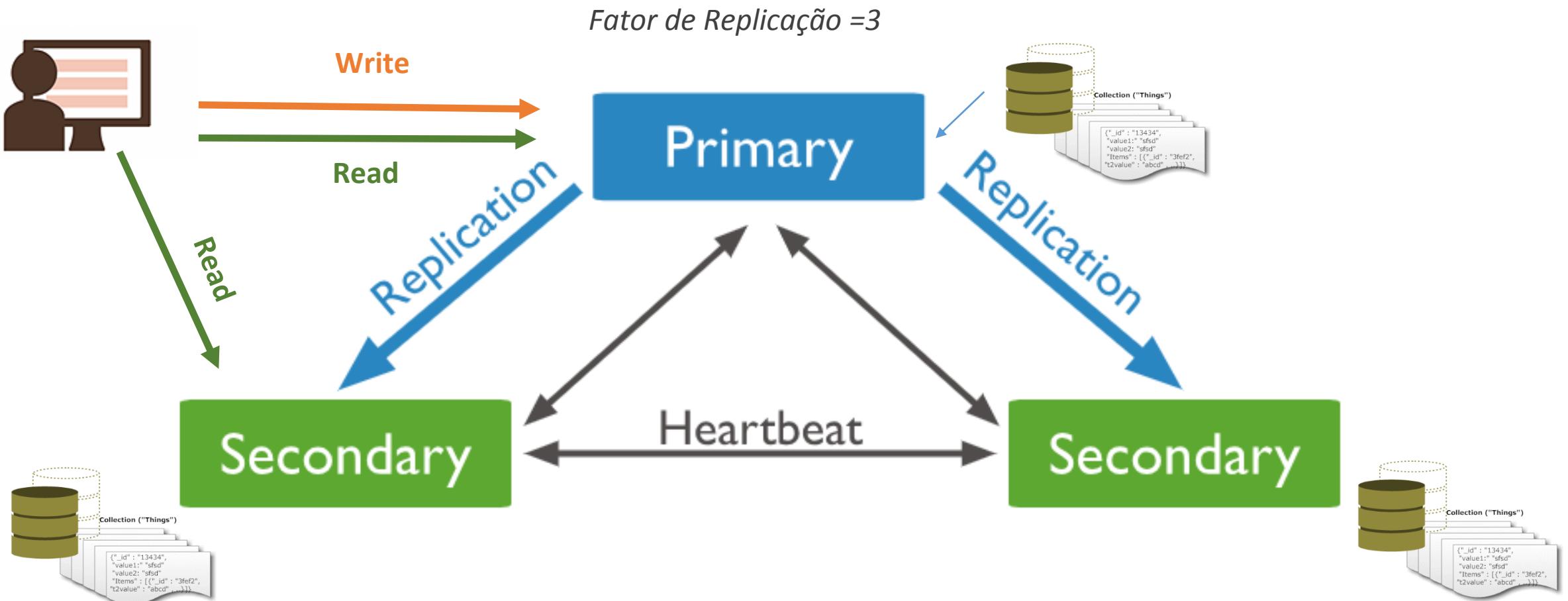
- MongoDB Replica Set.
- Demonstração.

# MongoDB Replica Set

- Replicação assíncrona;
- Replicação do *Oplog*;
- Escrita apenas no servidor primário;
- Réplicas secundárias permitem leituras.

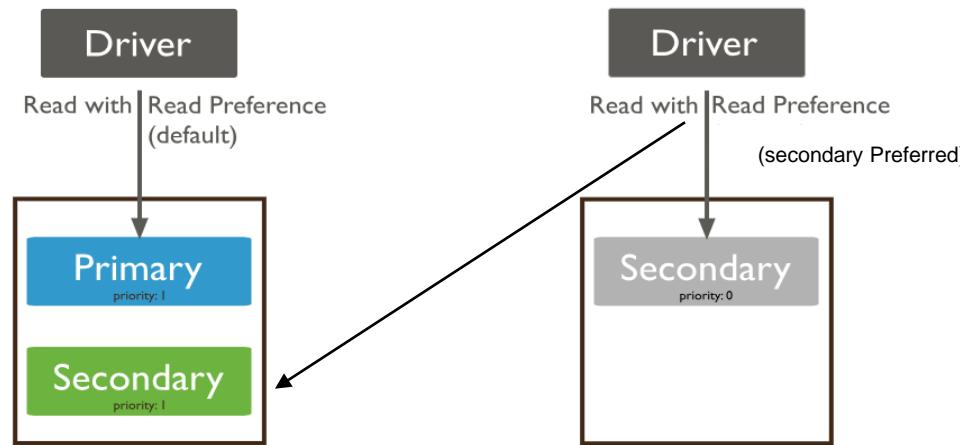
# MongoDB Replica Set

IGTI



# MongoDB Replica Set

- Leitura nas secundárias via parâmetro na string de conexão.



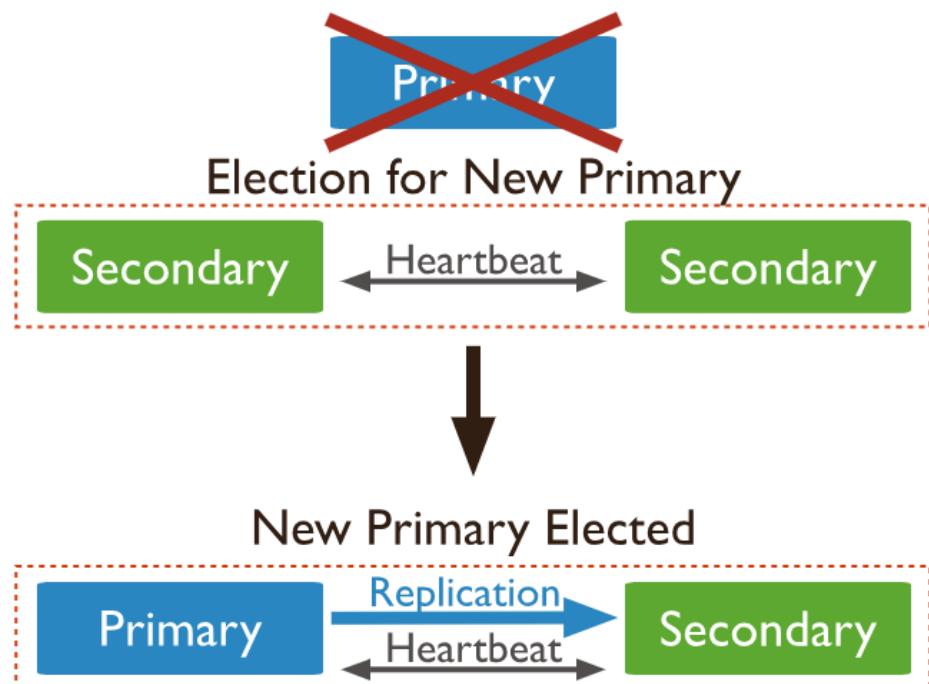
Tipo?

Hierárquico  
• Assíncrono

# MongoDB Replica Set

IGTI

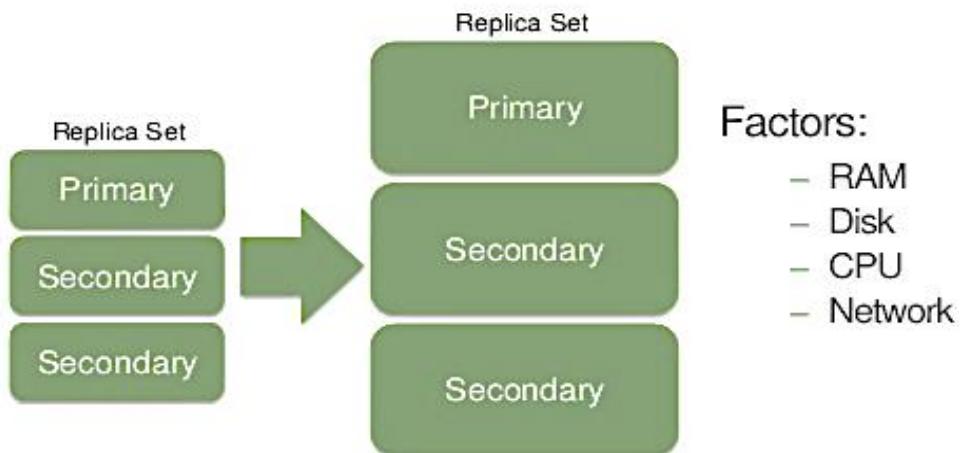
- Tolerância à partição de rede e falhas de hardware.



# MongoDB Replica Set

IGTI

- Escalabilidade vertical.



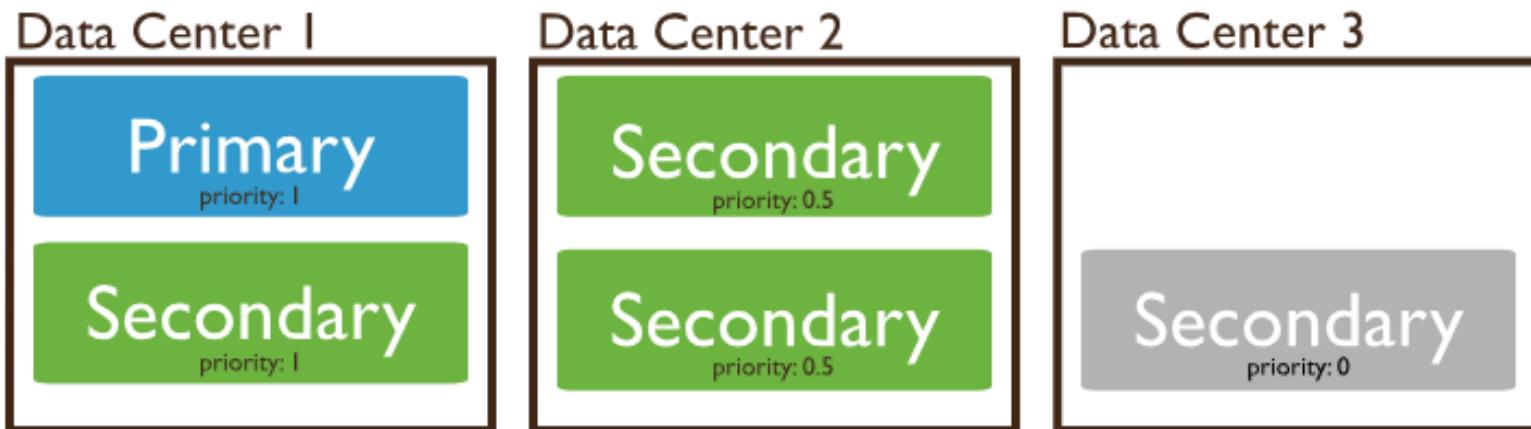
Factors:

- RAM
- Disk
- CPU
- Network

# MongoDB Replica Set

IGTI

- Ambientes de disaster recovery (DR).



# MongoDB Replica Set

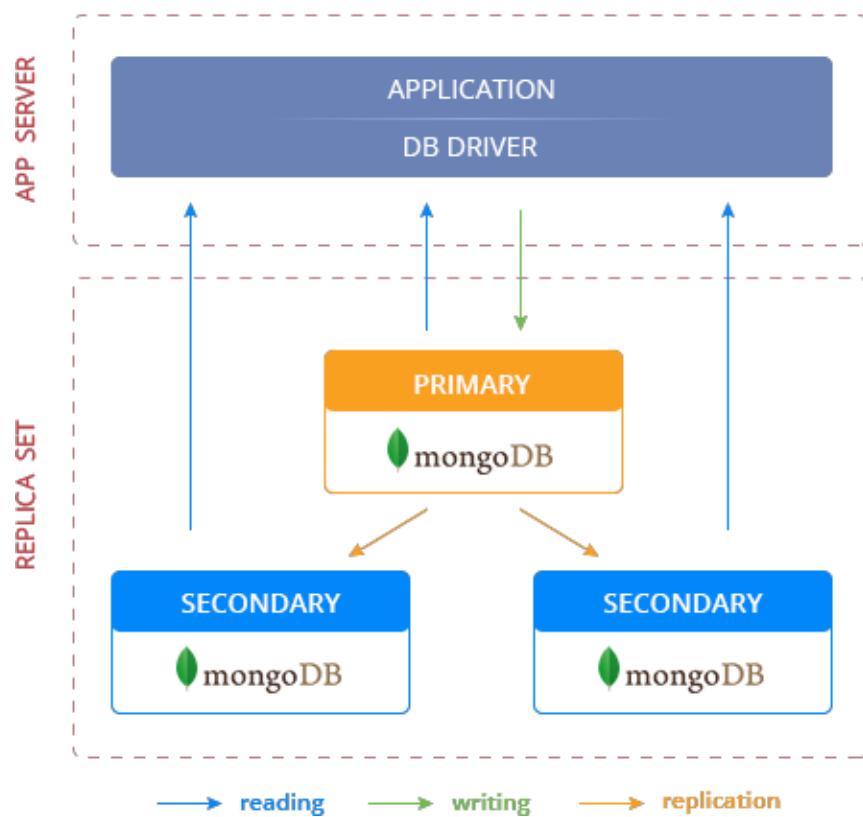
- Distribuição global de dados.



# MongoDB Replica Set

IGTI

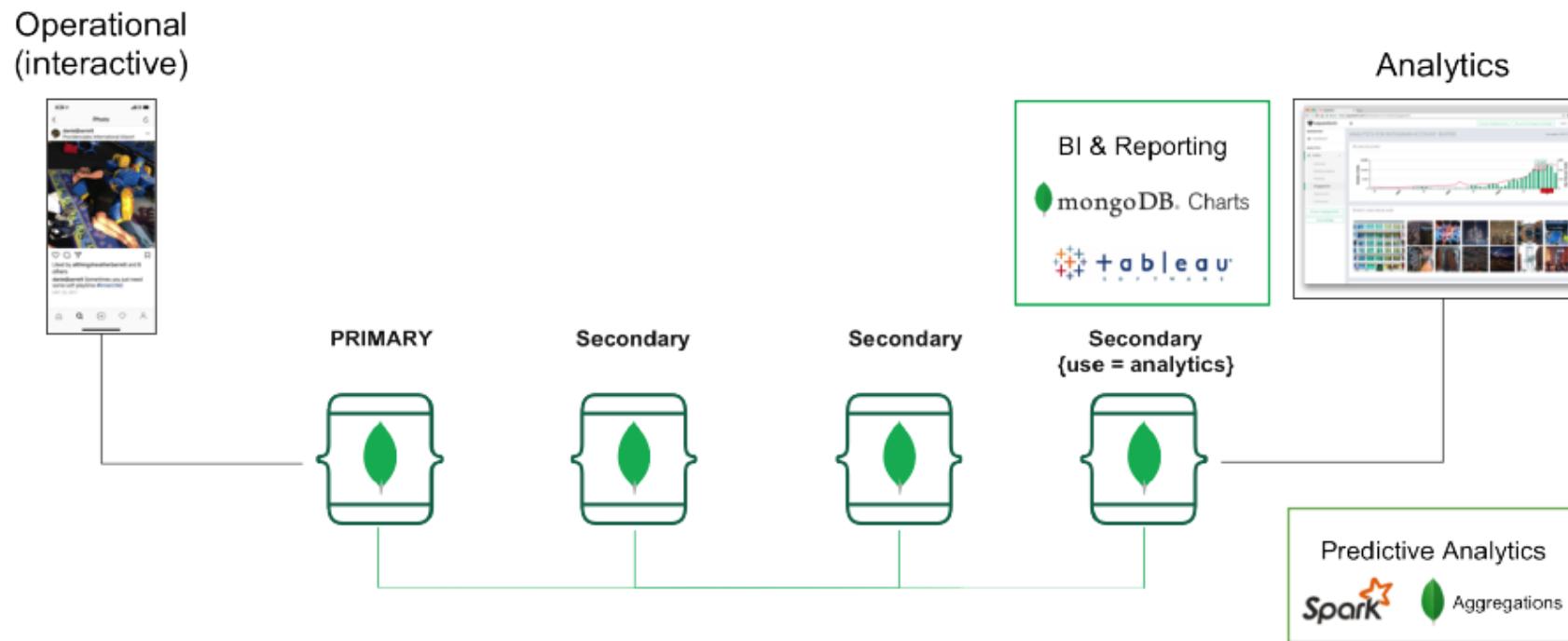
- Distribuição da carga de leituras.



# MongoDB Replica Set

IGTI

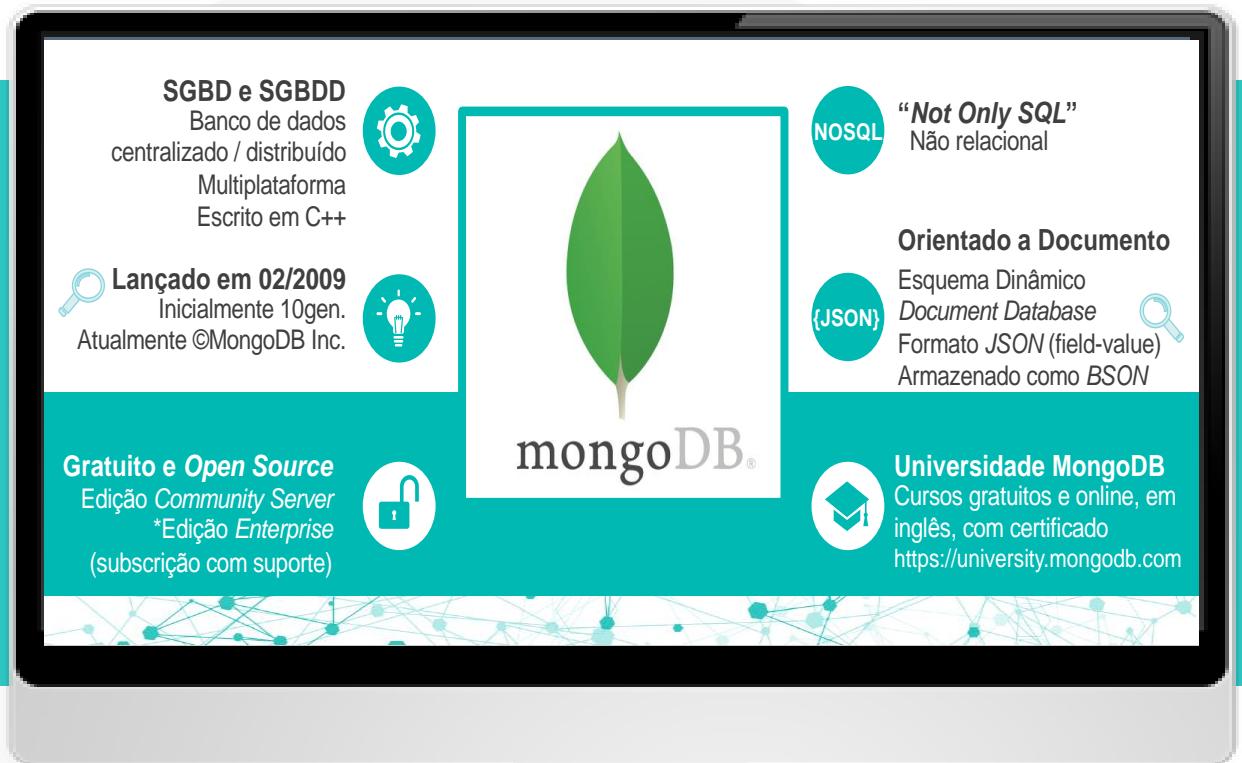
- Coexistência de workloads distintos, sem impactos para o sistema transacional.



# MongoDB Replica Set



 Demo



The infographic highlights MongoDB's features and ecosystem:

- SGBD e SGBDD**: Banco de dados centralizado / distribuído, Multiplataforma, Escrito em C++.
- Lançado em 02/2009**: Inicialmente 10gen. Atualmente ©MongoDB Inc.
- Gratuito e Open Source**: Edição Community Server (\*Edição Enterprise (subscrição com suporte))
- NOSQL**: “*Not Only SQL*” Não relacional
- Orientado a Documento**: Esquema Dinâmico, *Document Database*, Formato JSON (field-value), Armazenado como BSON
- (JSON)**: Formato JSON (field-value)
- Universidade MongoDB**: Cursos gratuitos e online, em inglês, com certificado <https://university.mongodb.com>

# Próxima Aula



❑ MongoDB Sharding.

# Distribuição de Dados no MongoDB

---

CAPÍTULO 4. AULA 4.2. MONGODB SHARDING

PROF. GUSTAVO AGUILAR

# Nesta Aula



- MongoDB Sharding.
- MongoDB Sharding Zones.

# MongoDB Sharding

IGTI

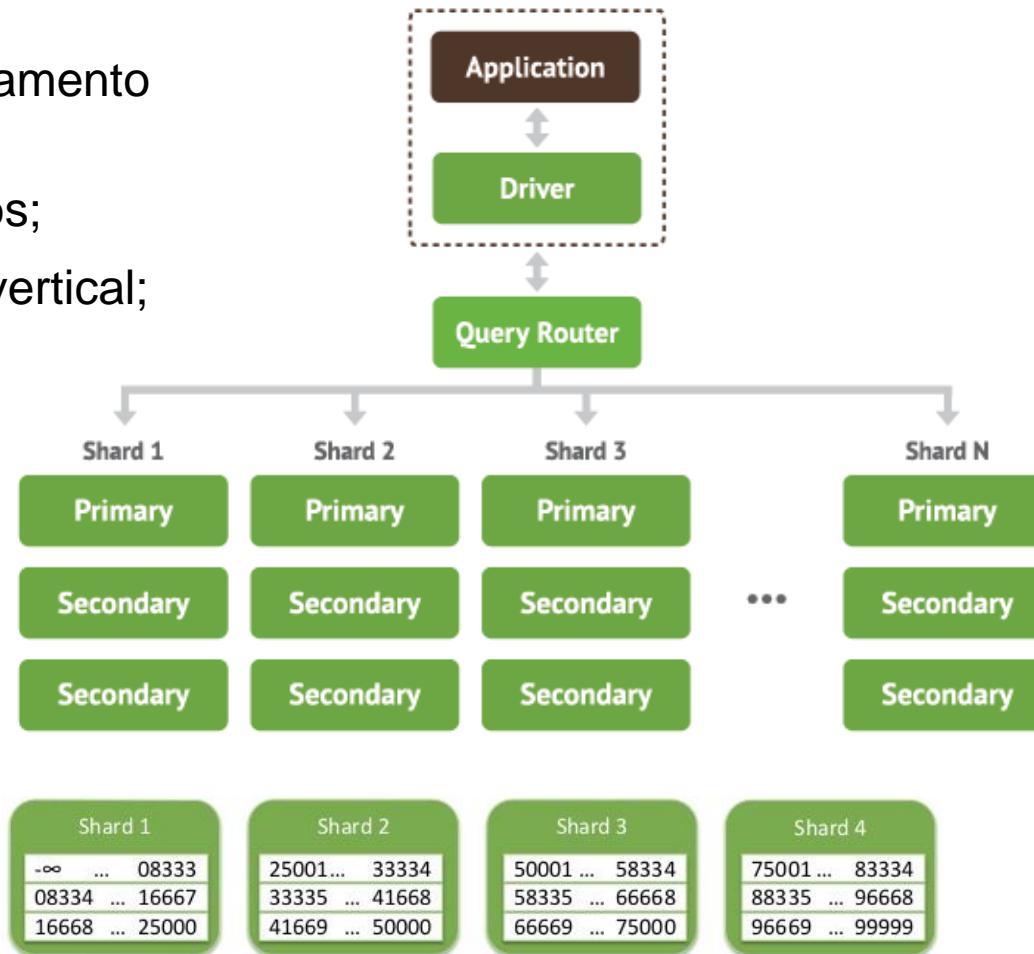
- Usa técnicas de particionamento horizontal (sharding);
- Leitura e escrita em todos nós;
- Escalabilidade: horizontal e vertical;
- Alta disponibilidade:
  - ✓ Replicação dos shards.

**Tipo?**

**Peer-to-Peer**

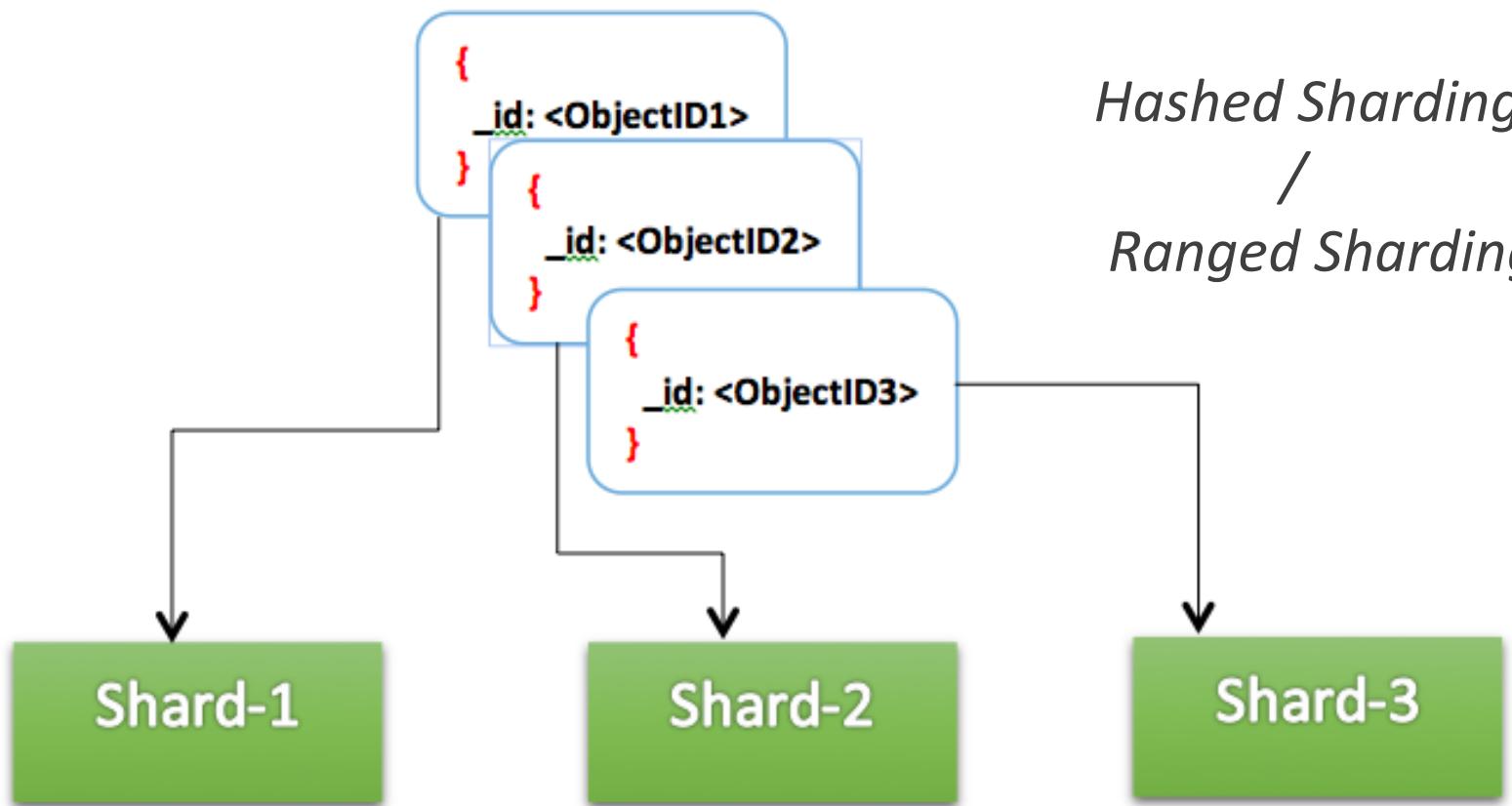
- Assíncrono

**Horizontal**



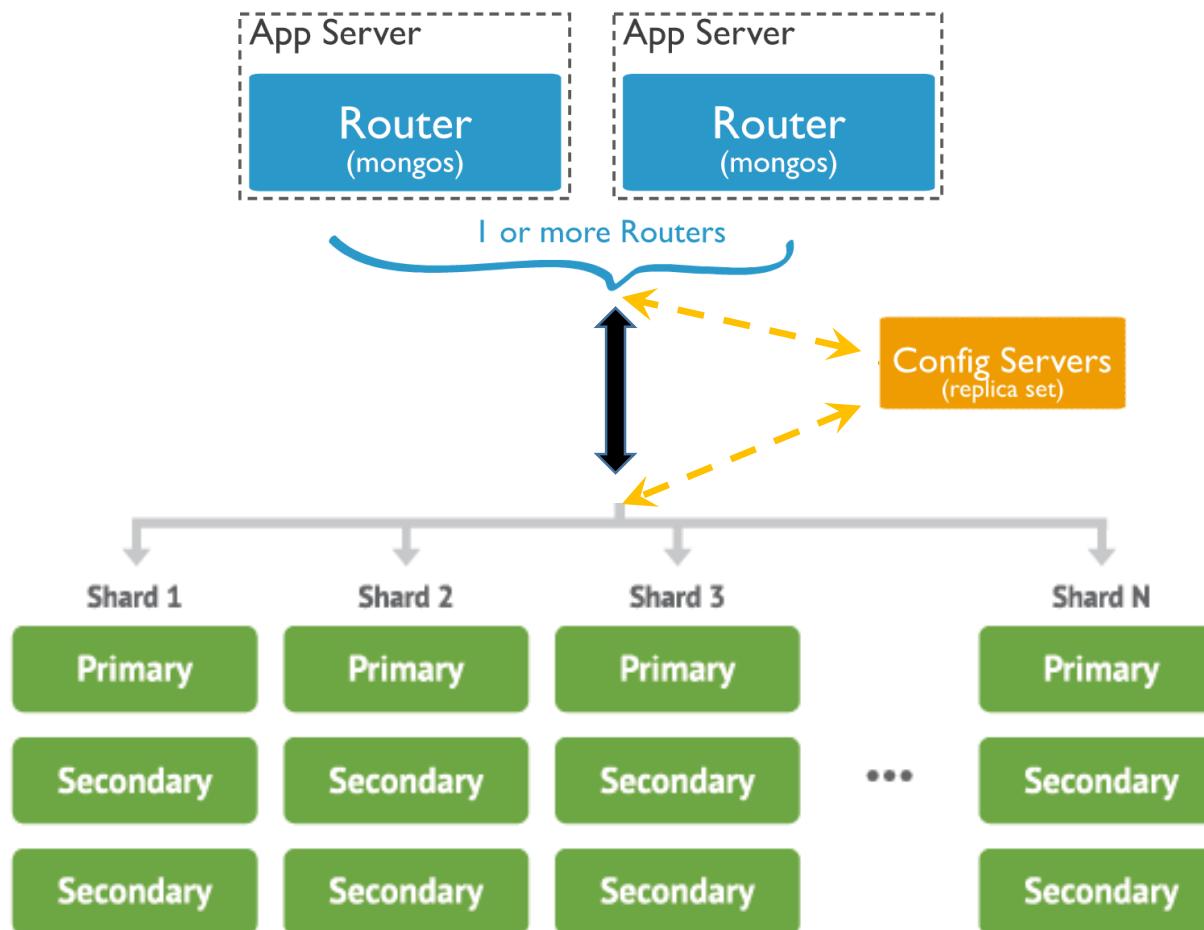
# MongoDB Sharding

IGTi



# MongoDB Sharding

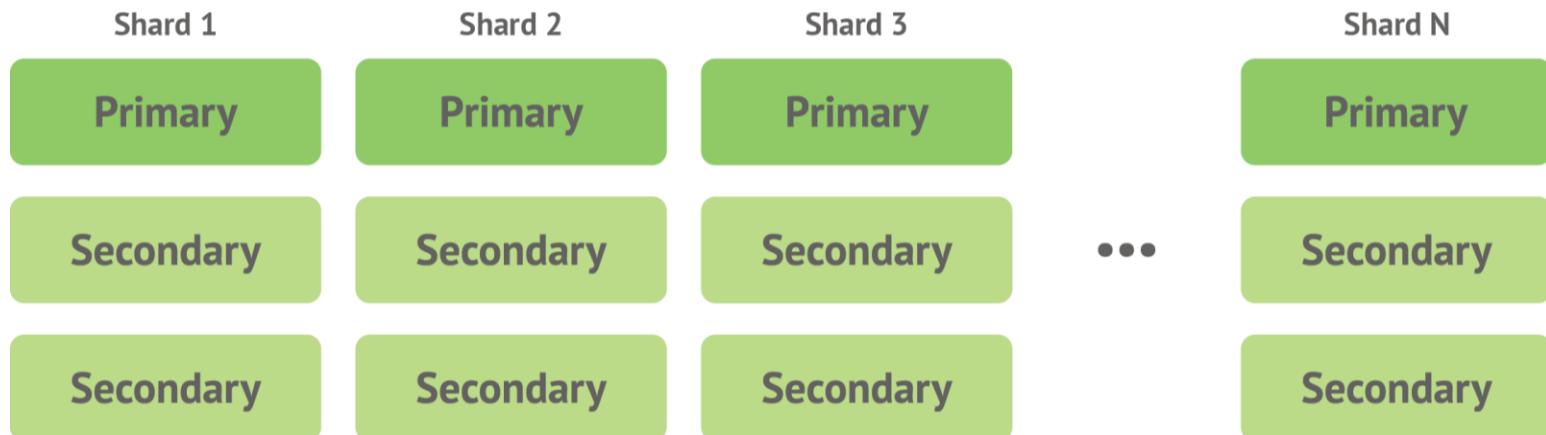
IGTi



# MongoDB Sharding

IGTI

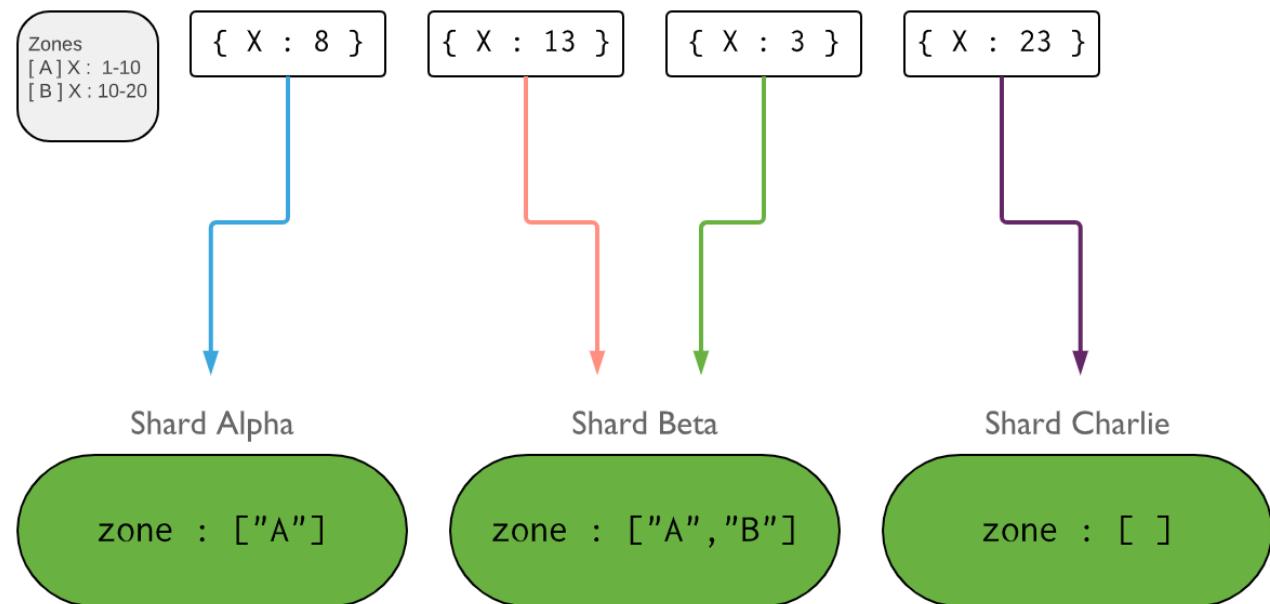
- Escalabilidade horizontal com sharding.



# MongoDB Sharding Zones

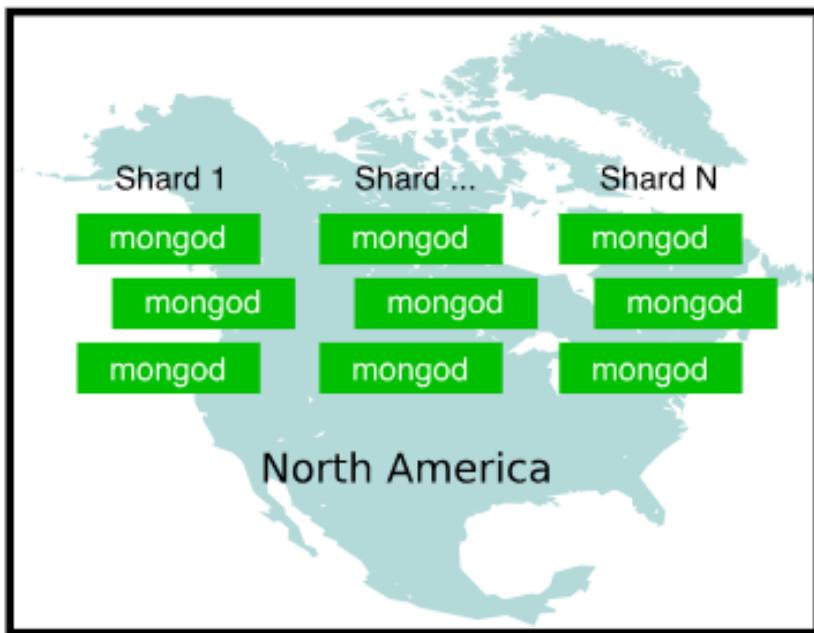
IGTi

- Multi-master.

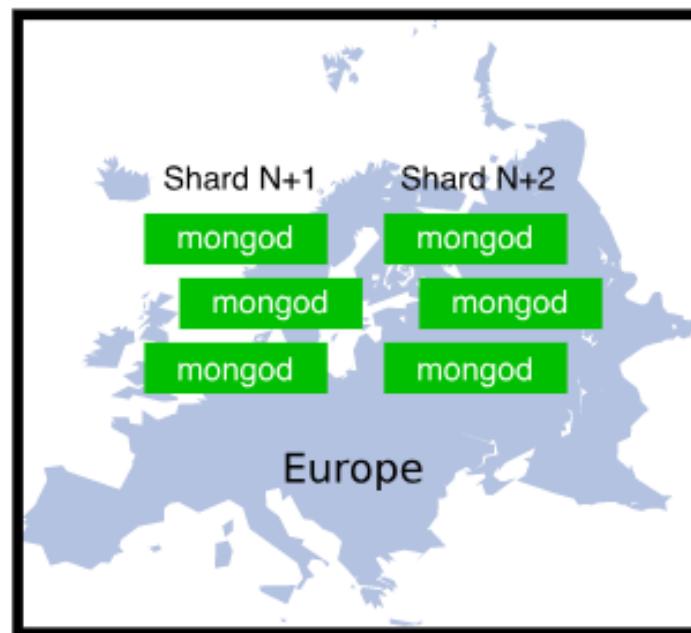


# MongoDB Sharding Zones

- Disaster recovery + otimização da performance de acesso aos dados + distribuição da carga.



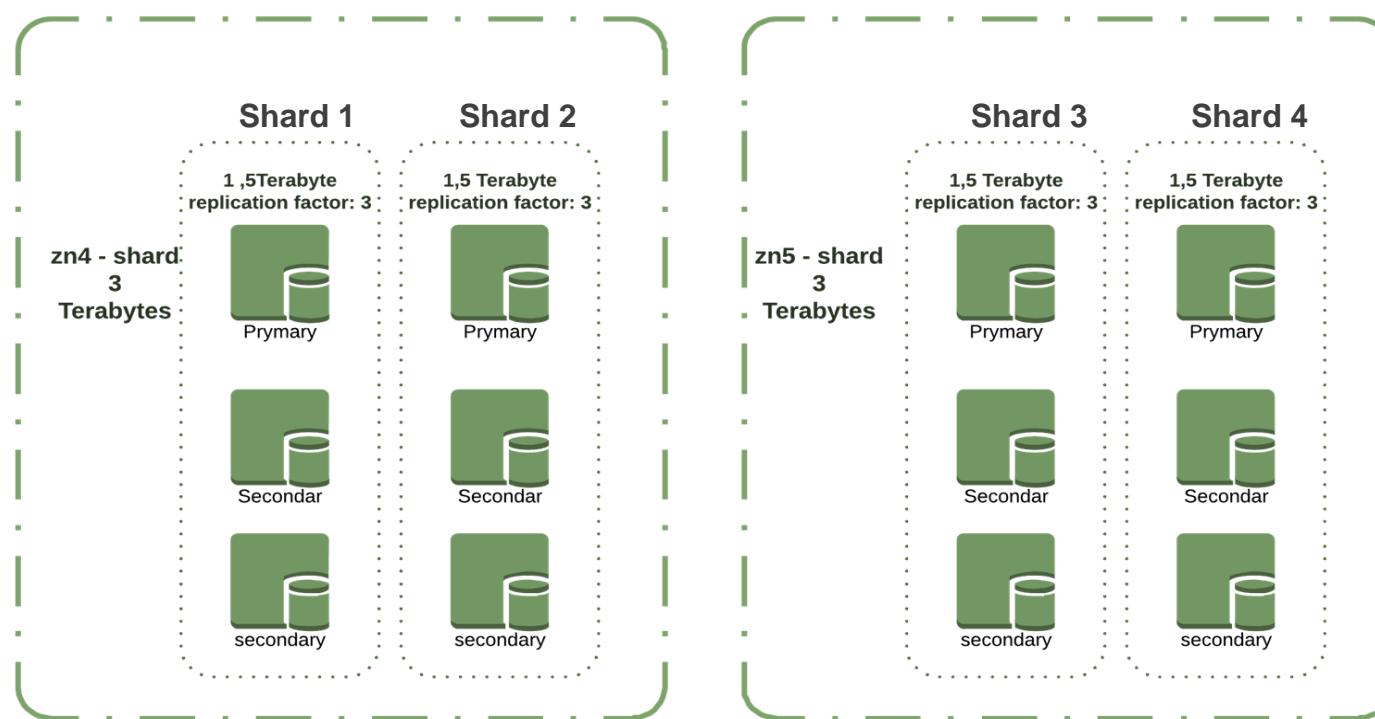
"NA" Zone



"EU" Zone

# MongoDB Sharding Zones

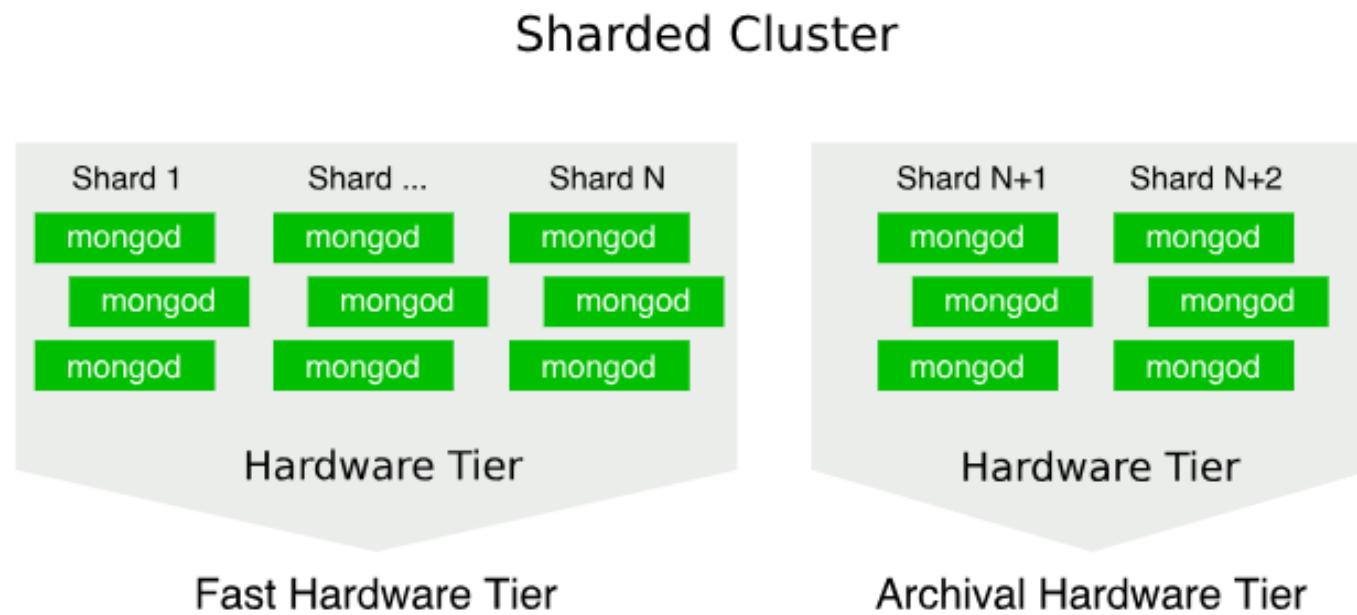
- Otimização de custos com a infra.



# MongoDB Sharding Zones

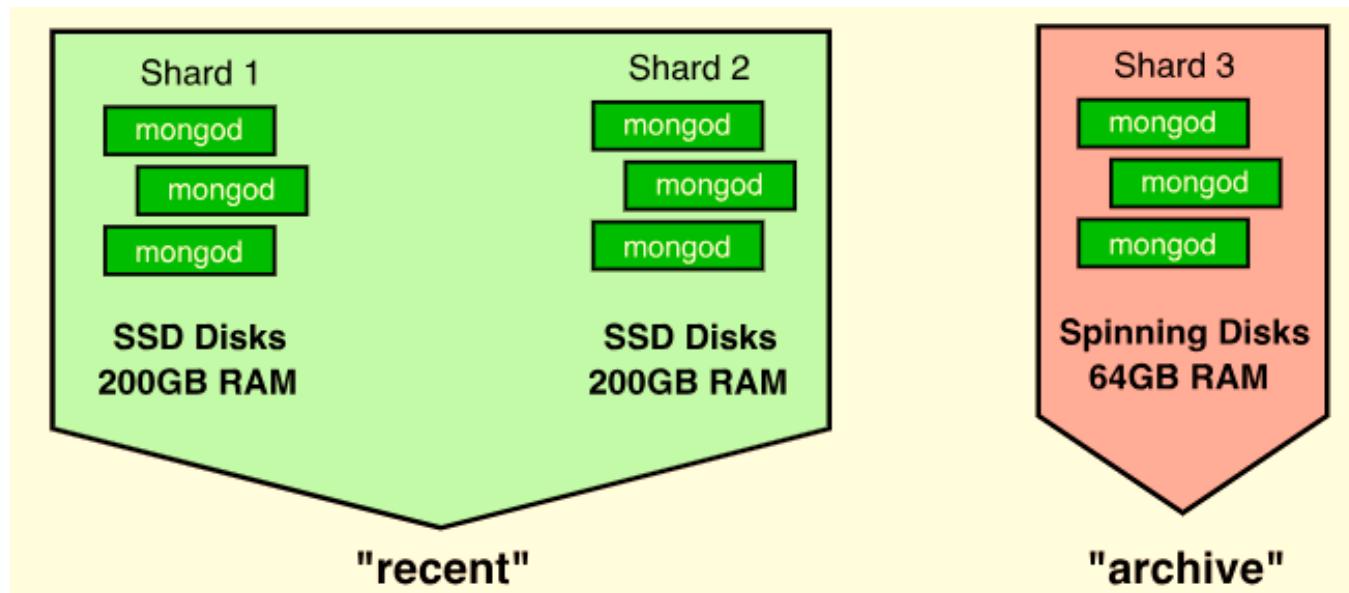
IGTI

- Otimização de custos com a infra.



# MongoDB Sharding Zones

- Arquivamento Automático Isolado com Sharding + Zones.



# Próxima Aula



MongoDB Atlas.

# Distribuição de Dados no MongoDB

---

CAPÍTULO 4. AULA 4.3. MONGODB ATLAS

PROF. GUSTAVO AGUILAR

# Nesta Aula



- MongoDB Atlas.
- Demonstração.

# MongoDB Atlas

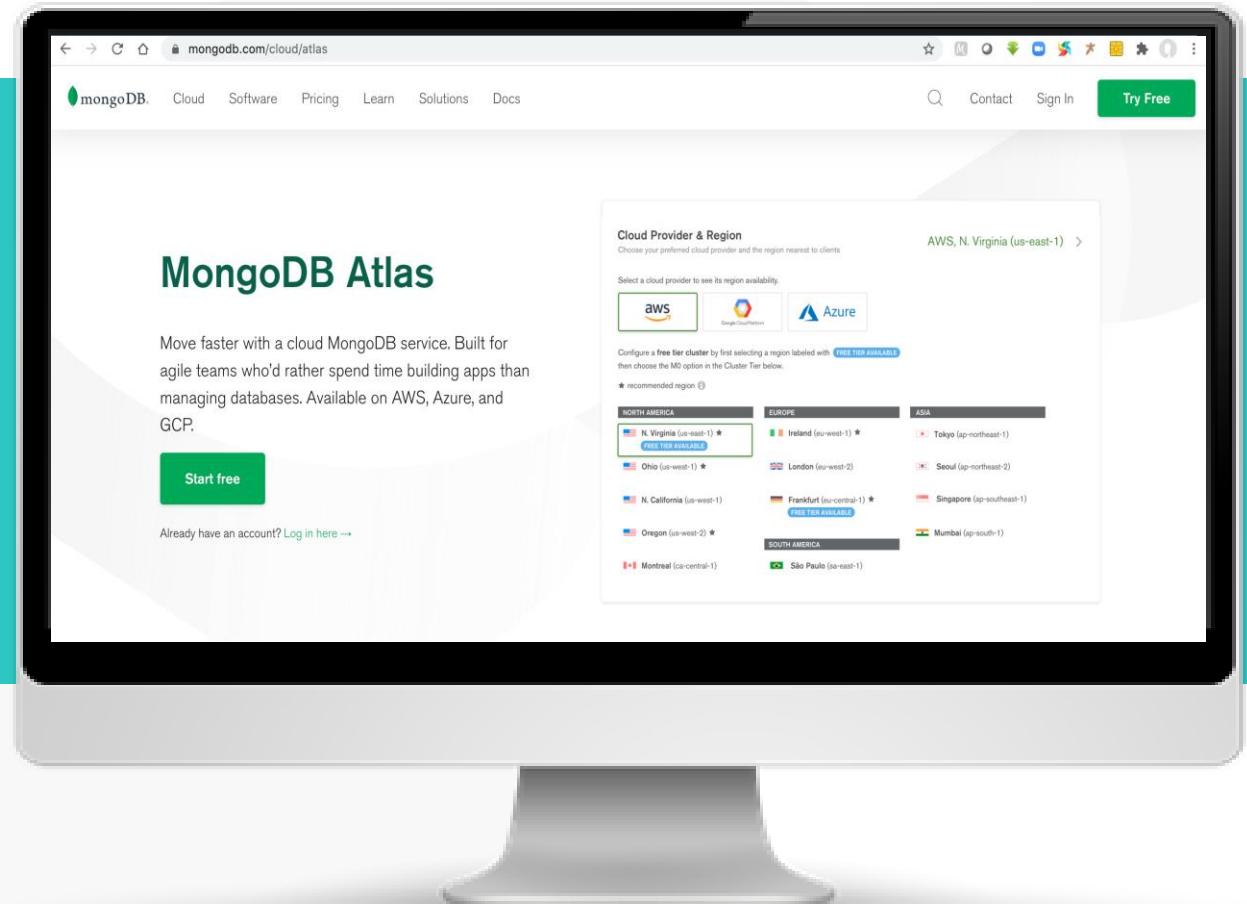


- Banco de Dados na nuvem, como serviço (PaaS);
- <https://www.mongodb.com/cloud/atlas>

The screenshot shows the MongoDB Atlas web interface. At the top, there's a navigation bar with links for 'Access Manager', 'Support', 'Billing', 'All Clusters', and a user profile for 'GUSTAVO A A'. Below the navigation is a secondary header with 'M001-MongoDB Basics' and 'Atlas' buttons, along with 'Realm' and 'Charts' links. On the left, a sidebar under 'DATA STORAGE' has 'Clusters' selected, with other options like 'Triggers', 'Data Lake', 'SECURITY', 'Database Access', 'Network Access', and 'Advanced'. The main content area is titled 'Clusters' and shows a 'Sandbox' section with a cluster named 'Cluster0' (Version 4.2.8). It includes tabs for 'CONNECT', 'METRICS', 'COLLECTIONS', and '...'. Below this, details show 'CLUSTER TIER' as 'M0 Sandbox (General)', 'REGION' as 'AWS / N. Virginia (us-east-1)', 'TYPE' as 'Replica Set - 3 nodes', and 'LINKED REALM APP' as 'None Linked'. A message at the bottom states 'Monitoring for Cluster0 is Paused' with a note that monitoring will resume when connected.

# MongoDB Atlas

 Demo



IGTI

# Próxima Aula



- Capítulo 5 - Aspectos Gerais de Performance em Banco de Dados.

# Performance e Otimização

**Capítulo 5. Aspectos Gerais de Performance em Banco de Dados**

**PROF. GUSTAVO AGUILAR**

# Aspectos Gerais de Performance em Banco de Dados

---

CAPÍTULO 5. AULA 5.1. ASPECTOS GERAIS DE INFRAESTRUTURA

PROF. GUSTAVO AGUILAR

# Nesta Aula



- ❑ Aspectos de Storage.
- ❑ Aspectos de Memória.
- ❑ Aspectos de CPU.

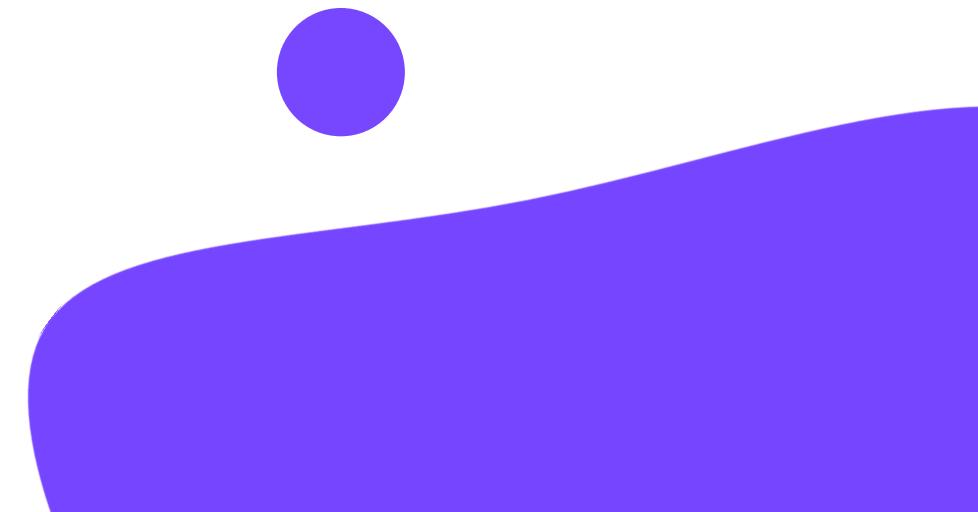
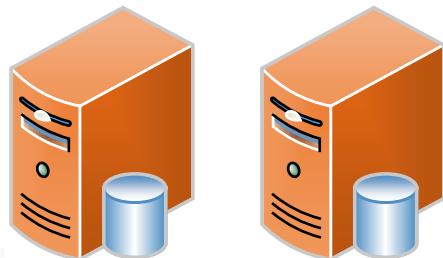
# Aspectos de Storage

- Tipo dos discos impacta diretamente na performance do banco de dados → Taxa de IOPS.

	3,5"	2,5"
5900 RPMs	50 IOPS	-
7200 RPMs	75 IOPS	95 IOPS
10000 RPMs	110 IOPS	140 IOPS
15000 RPMs	150 IOPS	180 IOPS
SSD	1500 IOPS	1500 IOPS

# Aspectos de Storage

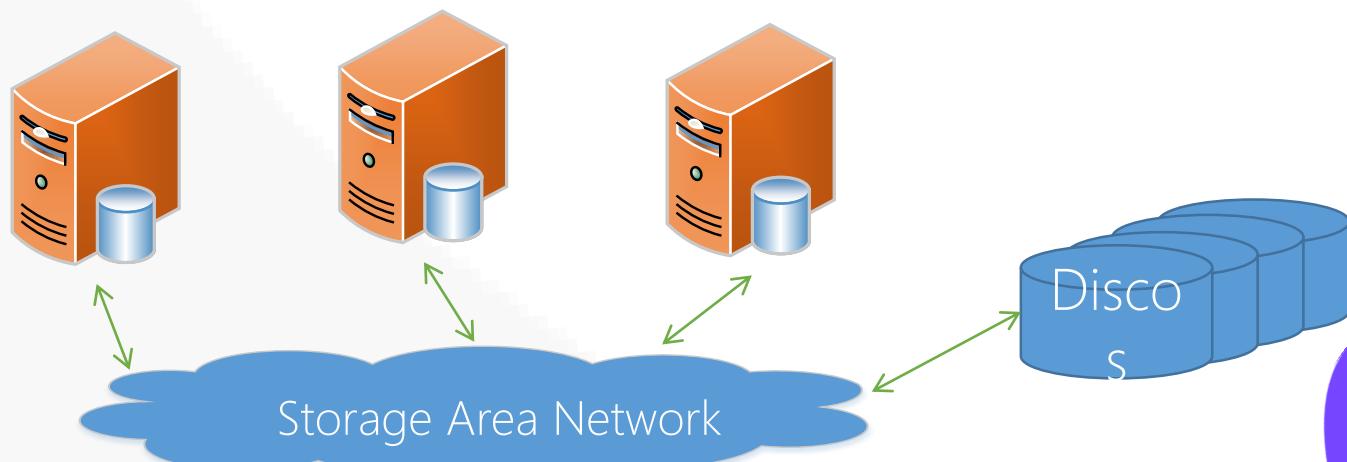
- Arquitetura / estratégia de alocação dos discos também podem impactar na performance do banco de dados.
- **Disco Atachado Diretamente no Servidor**
  - Boa performance;
  - “Ilhas” isoladas de storage nos servidores;
  - Baixa tolerância a falhas (exceto se for virtual);
  - Expansão fica limitada.



# Aspectos de Storage

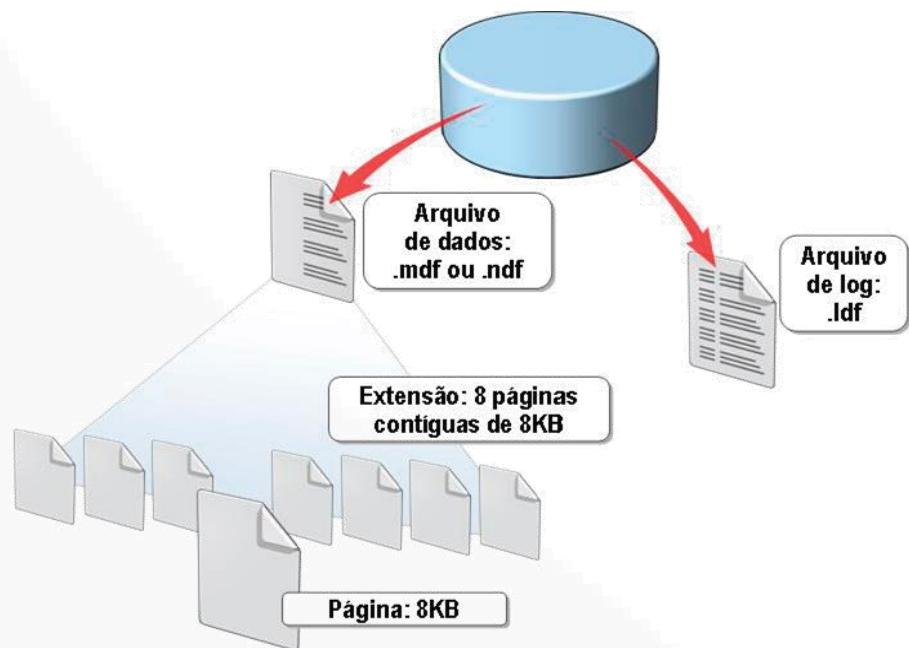
## ▪ Storage Area Network (SAN)

- Grande “ilha” de discos ligada ao parque de servidores → expansível;
- Rede dedicada que fornece acesso rápido aos discos;
- Incrementa a utilização da capacidade de armazenamento;
- Discos “virtuais” dedicados (não compartilhados com outro servidor).



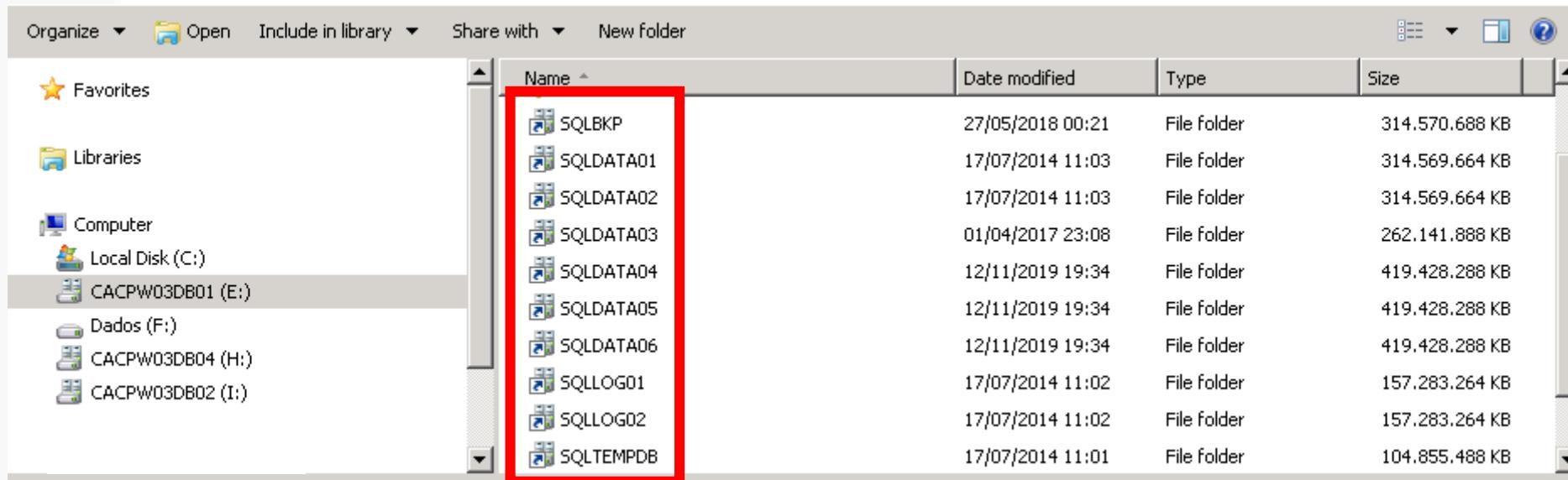
# Aspectos de Storage

- Formatação do disco impacta diretamente na performance.
  - SQL Server
    - NTFS com blocos de 64KB → 1 I/O = 1 extent ( $8 \times 8\text{KB} = 64\text{KB}$ ).



# Aspectos de Storage

- Use estratégias para facilitar a escalabilidade horizontal de storage.



Mount points  
(pontos de montagem)

# Aspectos de Storage

- Use estratégias para facilitar a escalabilidade horizontal de storage.

Filesystem	Size	Used	Avail	Use% Mounted on
/dev/sda1	9.8G	3.6G	5.7G	39% /
devtmpfs	16G	0	16G	0% /dev
tmpfs	16G	0	16G	0% /dev/shm
tmpfs	16G	1.5G	15G	10% /run
tmpfs	16G	0	16G	0% /sys/fs/cgroup
/dev/sda3	4.8G	36M	4.5G	1% /home
/dev/sda2	9.8G	1.6G	7.7G	16% /var
/dev/mapper/P40DIGIAvg-vol_mongodb	200	33M	200	1% /mongodb
/dev/mapper/P40DIGIAvg-vol_mongodb_log	500	170	340	33% /mongodb/log
/dev/mapper/vg_agents-vol_xvmanager	4.8G	2.4G	2.3G	52% /usr/local/xvmanager
/dev/mapper/vg_agents-vol_openv	9.8G	1.3G	8.0G	14% /usr/openv
/dev/mapper/vg_agents-vol_controls	4.8G	331M	4.3G	8% /controlM
/dev/mapper/P40DIGIAvg-vol_mongodb_data	2500	9.1G	241G	4% /mongodb/data
/dev/mapper/P40DIGIAvg-vol_mongodb_data_phonedirectory	2000	5.5G	195G	3% /mongodb/data/phonedirectory
/dev/mapper/P40DIGIAvg-vol_mongodb_data_transactionlog	50G	46M	50G	1% /mongodb/data/transactionlog
/dev/mapper/P40DIGIAvg-vol_mongodb_data_journal	50G	333M	50G	1% /mongodb/data/journal

Mount points  
(pontos de montagem)

# Aspectos de Storage



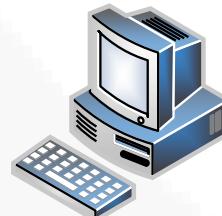
- Teste a performance dos discos do banco de dados antes!
  - Diskspd: <https://github.com/Microsoft/diskspd/releases/download/v2.0.21a/DiskSpd-2.0.21a.zip>
    - diskspd.exe -c16G -d300 -r -w30 -b8k -T4 -o8 -h -L c:\Test.dat

CPU	Usage	User	Kernel	Idle
0	43.48%	4.46%	39.02%	56.52%
1	17.48%	2.83%	14.65%	82.52%
2	17.87%	2.74%	15.12%	82.13%
3	18.85%	2.84%	16.01%	81.15%
avg.	24.42%	3.22%	21.20%	75.58%
Total IO	bytes	I/Os	MB/s	I/O per s
thread				
0	17239121920	2104385	54.80	7014.26
1	16413868032	2003646	52.18	6678.48
2	16521814016	2016823	52.52	6722.41
3	16364896256	1997668	52.02	6658.56
total:	66539700224	8122522	211.51	27073.71
AvgLat				1.179
LatStdDev				9.261
Read IO	bytes	I/Os	MB/s	I/O per s
thread				
0	12066045952	1472906	38.35	4909.44
1	11485511680	1402040	36.51	4673.23
2	11564048384	1411627	36.76	4785.19
3	11451154432	1397846	36.40	4659.25
total:	46566760448	5684419	148.02	18947.11
AvgLat				1.206
LatStdDev				8.887
Write IO	bytes	I/Os	MB/s	I/O per s
thread				
0	5173075968	631479	16.44	2104.82
1	4928356352	601606	15.67	2005.25
2	4957765632	605196	15.76	2017.22
3	4913741824	599822	15.62	1999.31
total:	19972939726	2438103	63.49	8126.60
AvgLat				1.115
LatStdDev				10.078

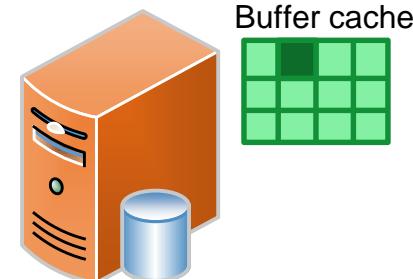
# Aspectos de Memória

- Normalmente, banco de dados usa mais memória (cache) que CPU;
- Projeção baseada na relação:
  - ***Total de Storage para Dados x Memória RAM para SGBD.***

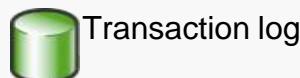
1 Query enviada pelo usuário / app.



2 SGBD carrega / modifica os dados em cache



3 Ocorre o flush dos dados modificados no cache.



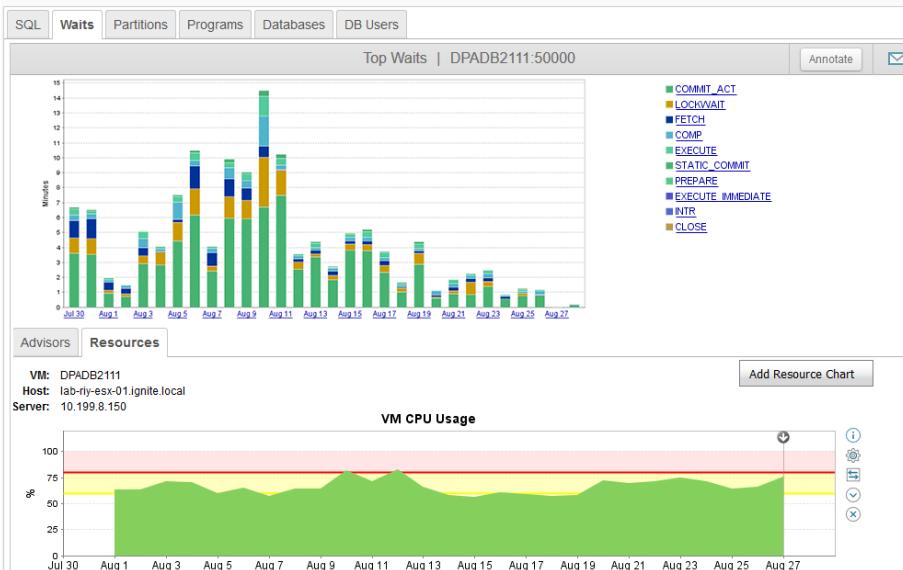
4 Páginas são escritas no(s) arquivo(s) de dados.



# Aspectos de CPU



- Quantidade de CPU x Paralelismo Desejado;
- Alto consumo constante de CPU → normalmente está relacionado à falta de índice, fragmentação, estatística desatualizada, ou excesso de concorrência / paralelismo em detrimento do número de CPUs.



# Conclusão



- ✓ Recurso é finito:
  - ✓ Dimensionar bem para o projeto;
  - ✓ Planejar o Crescimento Vegetativo (CVT);
  - ✓ Acompanhar.
- ✓ Queries ineficientes podem ser compensadas quando há recursos de sobra, até que...
- ✓ Queries eficientes sendo impactadas por escassez de recursos / topologia da solução:
  - ✓ Ausência de escalabilidade horizontal;
  - ✓ Estrutura / formatação dos discos;
  - ✓ Quantidade de CPU e memória.

# Próxima Aula



- Aspectos Gerais dos SGBDs.

# Aspectos Gerais de Performance em Banco de Dados

---

CAPÍTULO 5. AULA 5.2. ASPECTOS GERAIS DOS SGBDS

PROF. GUSTAVO AGUILAR

# Nesta Aula



- ❑ Aspectos da Instância de Banco de Dados.
- ❑ Aspectos da Estrutura do Banco de Dados.
- ❑ Aspectos da Criação de Tabelas.

# Aspectos da Instância

- Configuração correta dos parâmetros que afetam diretamente a performance dos banco de dados, como por exemplo:
  - ✓ Memória: mínimo e máximo;
  - ✓ CPU: usar todas? Afinidade de CPU?
  - ✓ Paralelismo: ativado / desativado? Métrica para parallelizar?
  - ✓ Número máximo de sessões abertas;
  - ✓ Quantidade máxima de sessões concorrentes;
  - ✓ Configurar algum mecanismo de gerenciamento dos recursos internos da instância entre os bancos (Resource Governor)?

# Aspectos da Instância

- Cada instância com seu conjunto de discos.

Name	Date modified	Type	Size
SQLBKP	27/05/2018 00:21	File folder	314.570.688 KB
SQLDATA01	17/07/2014 11:03	File folder	314.569.664 KB
SQLDATA02	17/07/2014 11:03	File folder	314.569.664 KB
SQLDATA03	01/04/2017 23:08	File folder	262.141.888 KB
SQLDATA04	12/11/2019 19:34	File folder	419.428.288 KB
SQLDATA05	12/11/2019 19:34	File folder	419.428.288 KB
SQLDATA06	12/11/2019 19:34	File folder	419.428.288 KB
SQLLOG01	17/07/2014 11:02	File folder	157.283.264 KB
SQLLOG02	17/07/2014 11:02	File folder	157.283.264 KB
SQLTEMPDB	17/07/2014 11:01	File folder	104.855.488 KB

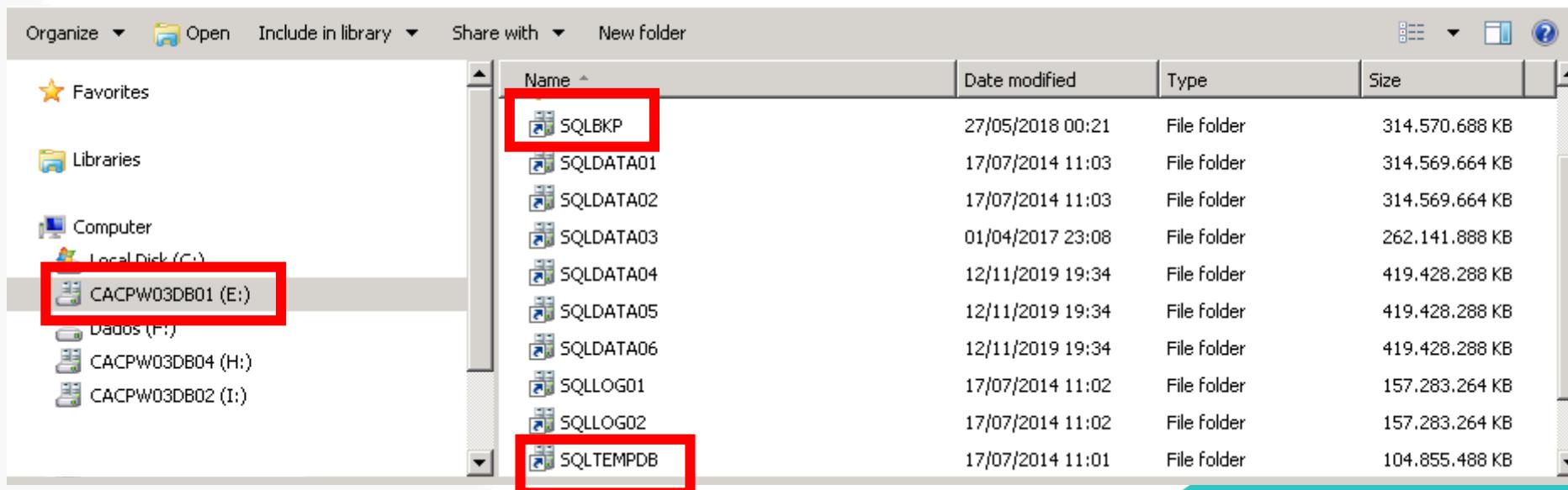
# Aspectos da Instância

- Cada instância com discos separados para cada recurso.

Filesystem	Size	Used	Avail	Use% Mounted on
/dev/sda1	9.8G	3.6G	5.7G	39% /
devtmpfs	16G	0	16G	0% /dev
tmpfs	16G	0	16G	0% /dev/shm
tmpfs	16G	1.5G	15G	10% /run
tmpfs	16G	0	16G	0% /sys/fs/cgroup
/dev/sda3	4.8G	36M	4.5G	1% /home
/dev/sda2	9.8G	1.6G	7.7G	16% /var
/dev/mapper/P40DIGIAvg-vol_mongodb	20G	33M	20G	1% /mongodb
/dev/mapper/P40DIGIAvg-vol_mongodb_log	50G	17G	34G	33% /mongodb/log
/dev/mapper/vg_agents-vol_lvmanager	4.8G	2.4G	2.3G	52% /usr/local/lvmanager
/dev/mapper/vg_agents-vol_openv	9.8G	1.3G	8.0G	14% /usr/openv
/dev/mapper/vg_agents-vol_controlm	4.8G	331M	4.3G	8% /controlm
/dev/mapper/P40DIGIAvg-vol_mongodb_data	250G	9.1G	241G	4% /mongodb/data
/dev/mapper/P40DIGIAvg-vol_mongodb_data_phonedirectory	200G	5.5G	195G	3% /mongodb/data/phonedirectory
/dev/mapper/P40DIGIAvg-vol_mongodb_data_transactionlog	50G	46M	50G	1% /mongodb/data/transactionlog
/dev/mapper/P40DIGIAvg-vol_mongodb_data_journal	50G	333M	50G	1% /mongodb/data/journal

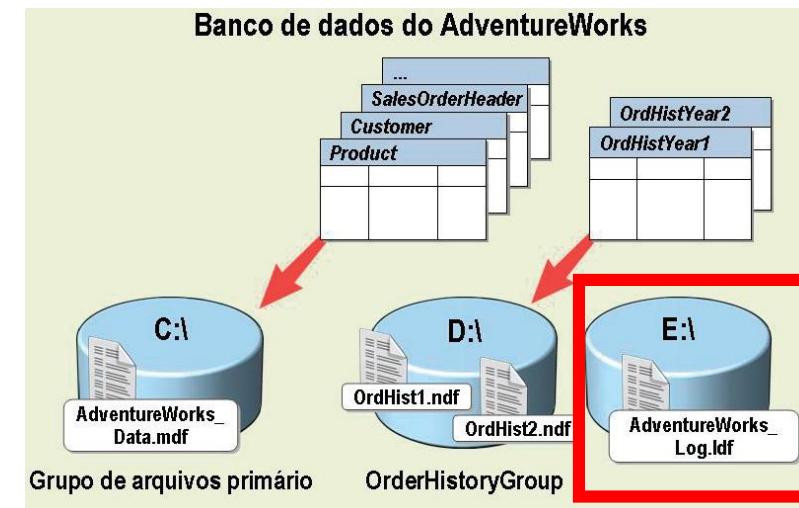
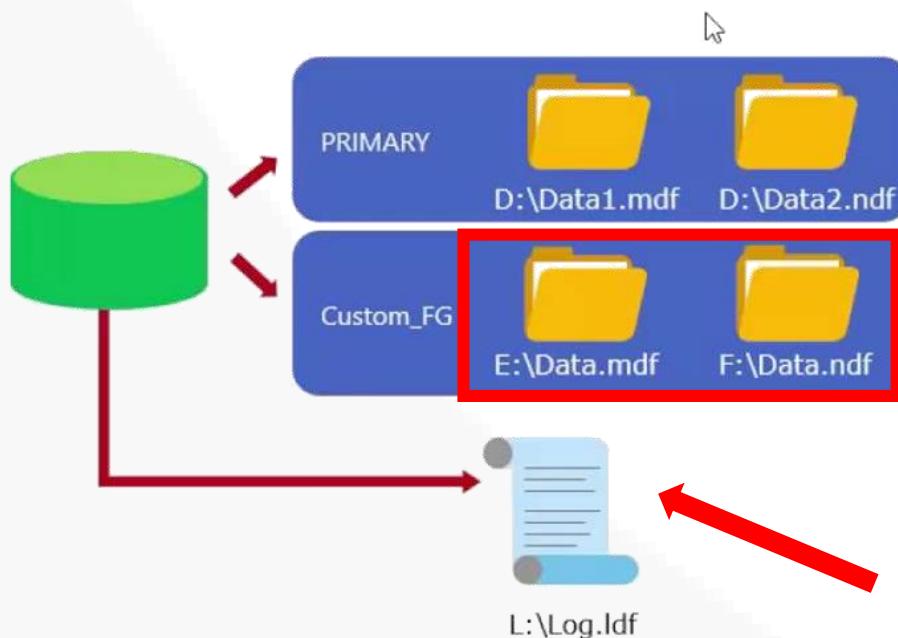
# Aspectos da Estrutura do BD

- Separação dos bancos de dados de Sistema (dicionário de dados do SGBD) dos bancos de usuário (aplicação).



# Aspectos da Estrutura do BD

- Separação das estruturas peculiares de cada banco de dados, em discos distintos.



Arquivo de log único  
por banco SQL Server

# Aspectos da Estrutura do BD

- Separação das estruturas peculiares de cada banco de dados, em discos distintos.

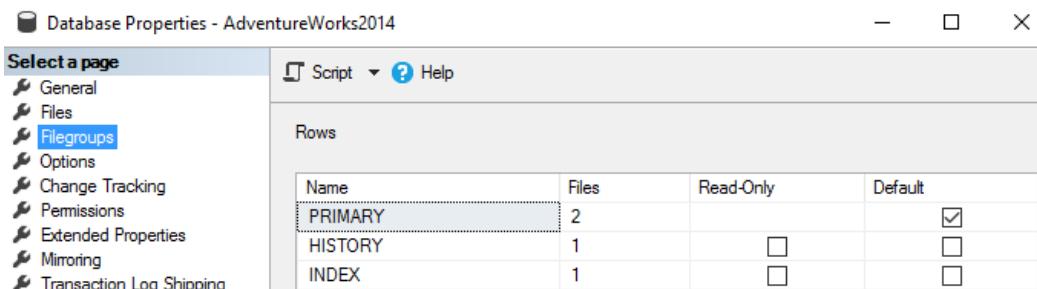
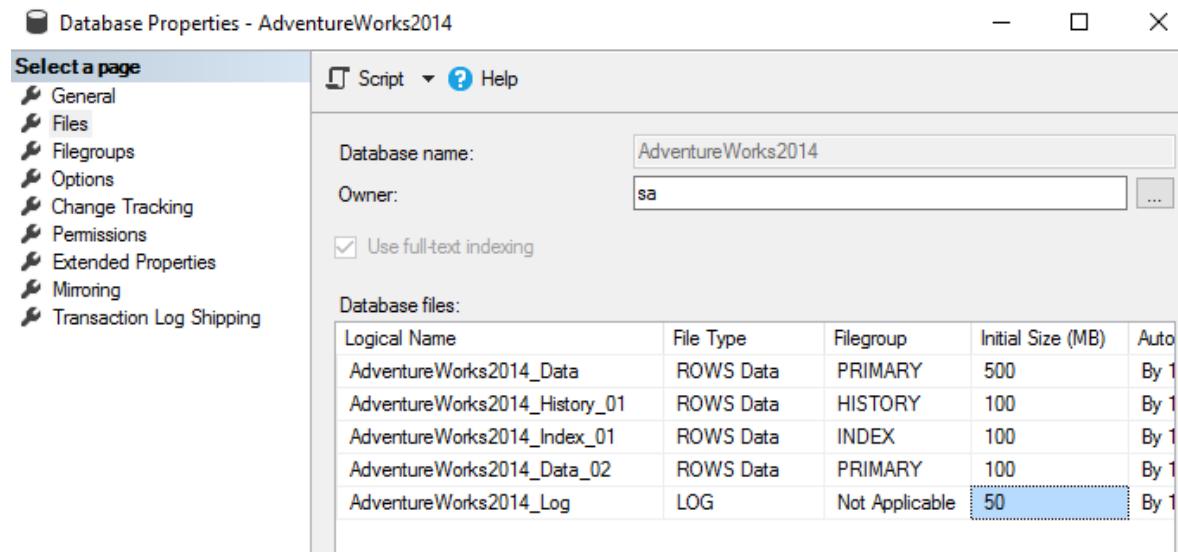
```
-bash-4.2$ cd /mongodb/data          *Parâmetro: directoryPerDB
-bash-4.2$ ls -l
total 356
drwx----- 4 mongod mongod   37 Aug 27  2019 admin
drwx----- 4 mongod mongod   37 Aug 27  2019 config
drwx----- 2 mongod mongod 4096 Aug 29 13:58 diagnostic.data
drwxr-xr-x 2 mongod mongod  110 Aug 22 22:59 journal
drwx----- 4 mongod mongod   37 Aug 27  2019 local
-rw----- 1 mongod mongod 45056 Jul 30 22:54 _mdb_catalog.wt
-rw----- 1 mongod mongod     7 Jul  9 22:54 mongod.lock
drwx----- 4 mongod mongod   37 Jul  9 22:54 phonedirectory
-rw----- 1 mongod mongod 36864 Aug 29 13:58 _storage.bson
-rw----- 1 mongod mongod  114 Aug 27  2019 storage.bson
drwx----- 4 mongod mongod   37 Jul 14 15:11 transactionlog
-rw----- 1 mongod mongod   48 Aug 27  2019 wirecursor
-rw----- 1 mongod mongod 4096 Jul  9 22:53 WiredTigerLAS.wt
-rw----- 1 mongod mongod   21 Aug 27  2019 WiredTiger.lock
-rw----- 1 mongod mongod 1071 Aug 29 13:58 WiredTiger.turtle
-rw-r----- 1 mongod mongod 253952 Aug 29 13:58 WiredTiger.wt
-bash-4.2$
```

```
-bash-4.2$ cd /mongodb/data/phonedirectory
-bash-4.2$ ls -l
total 4          *Parâmetro: directoryForIndexes
drwx----- 2 mongod mongod 303 Mar  6 17:09 collection
drwx----- 2 mongod mongod 4096 Mar  6 17:09 index
```

# Aspectos da Criação de Tabelas

IGTI

- Segregar tabelas e índices.



# Aspectos da Criação de Tabelas



- Segregar tabelas e índices.

```
CREATE TABLE [Person].[EmailAddress]
(
    [BusinessEntityID] [int] NOT NULL,
    [EmailAddressID] [int] IDENTITY(1,1) NOT NULL,
    [EmailAddress] [nvarchar](50) NULL,
    [rowguid] [uniqueidentifier] ROWGUIDCOL NOT NULL,
    [ModifiedDate] [datetime] NOT NULL,
    CONSTRAINT [PK_EmailAddress_BusinessEntityID_EmailAddressID] PRIMARY KEY CLUSTERED
    (
        [BusinessEntityID] ASC,
        [EmailAddressID] ASC
    )
    ON [PRIMARY]
)
ON [PRIMARY]
GO
```

```
CREATE NONCLUSTERED INDEX [IX_EmailAddress_EmailAddress] ON [Person].[EmailAddress]
(
    [EmailAddress] ASC
)
ON [INDEX]
GO
```

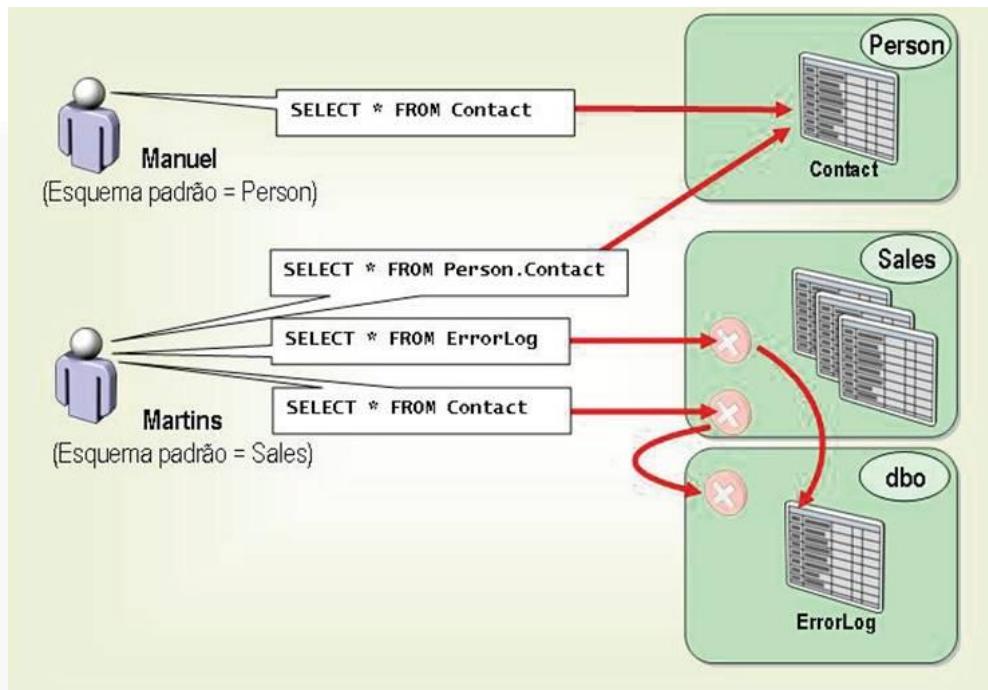
# Aspectos da Criação de Tabelas



- Usar compressão de dados na tabela?
  - De página?
  - De linhas?
- Tabela em memória?
- Tabela particionada?
- Tipo de tabela x Meio de armazenamento está adequado?
- Rotina de manutenção da tabela?
  - Desfragmentação;
  - Atualização de estatísticas.

# Aspectos da Criação de Tabelas

- Schema x Default schema dos usuários x Queries



# Próxima Aula



- Índices.

# Aspectos Gerais de Performance em Banco de Dados

---

CAPÍTULO 5. AULA 5.3. ÍNDICES

PROF. GUSTAVO AGUILAR

# Nesta Aula



- ❑ Introdução à Índices.
- ❑ Tipos de Índices.
- ❑ Shift Index.
- ❑ Efeitos Colaterais dos Índices.

# Introdução à Índices

- A palavra **índice** vem do latim *índex* → “o que **indica**”.
- Em um **livro**: o índice é uma lista ordenada dos capítulos e sessões que essa publicação contém, com o **respectivo local** (número da página) onde eles podem ser encontrados.

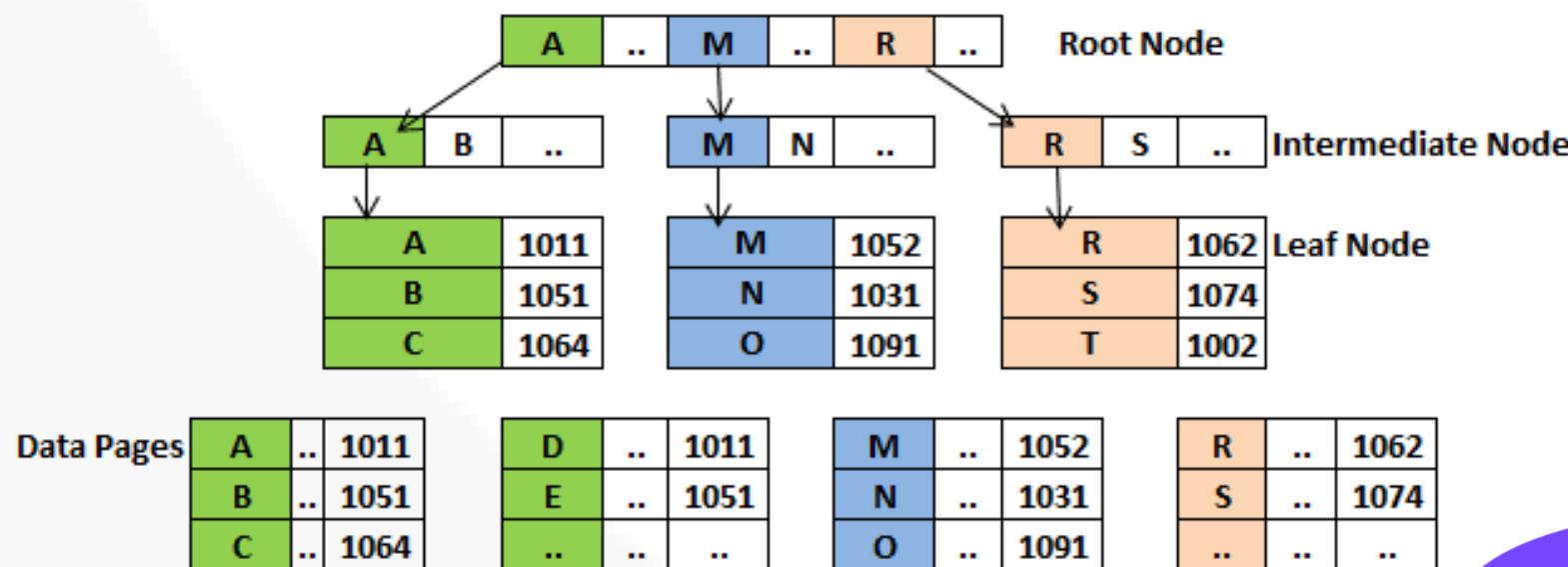
## Índice Remissivo

<b>A</b>		<b>K</b>		<b>Q</b>	
Alef .....	3	Kepler.....	2	Quociente .....	3
Análise.....	1				
<b>B</b>		<b>L</b>		<b>R</b>	
Bola .....		Limite.....	2	Razão .....	3
aberta .....	3	infinito.....	4	Riemman.....	4
fechada .....	4				
<b>F</b>		<b>M</b>		<b>S</b>	
Função .....	1	Matemática .....	2	Somatório .....	3
<b>H</b>		<b>N</b>		<b>T</b>	
História .....		Napier.....	2	Polinômios .....	2
da Matemática	1			Topologia .....	3

- Função do índice é reduzir o tempo gasto para localizar conteúdo.

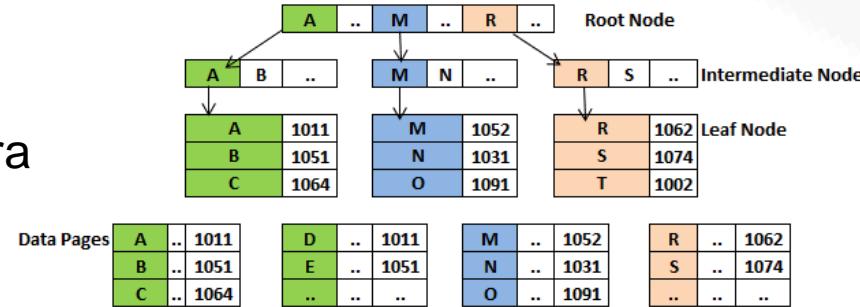
# Introdução à Índices

- Em SGBD: índice é uma **estrutura auxiliar** associada a uma tabela, geralmente organizada como uma árvore binária.



# Introdução à Índices

- Consiste de uma estrutura com **ponteiros para os dados armazenados nas colunas da tabela**;
- SGBD utiliza essa estrutura (o índice) de maneira semelhante ao índice remissivo de um livro:
  - Consulta-o primeiramente para encontrar determinado assunto;
  - Localiza a sua posição em uma determinada página;
  - Vai até essa página e lê os dados.
- Objetivo é acelerar o tempo de execução de uma consulta → aumentar a performance.



# Introdução à Índices

Índice para a coluna Name

Name	Reference
James	1
Lyon	2
Pat	3
Philip	7
Ramel	4
Ramel	6
Ramel	8
Ramel	1000
Rozer	9
Tim	5
...	...
...	...

Tabela com todas as colunas

Row	Name	Age
1	James	23
2	Lyon	22
3	Pat	20
4	Ramel	23
5	Tim	20
6	Ramel	21
7	Philip	19
8	Ramel	22
9	Rozer	17
..	...	...
..	...	...
1000	Ramel	19

pointing table row  
pointing table row  
pointing table row  
pointing table row  
pointing table row

\*Sem a existência do índice → **FULL TABLE SCAN.**

# Introdução à Índices

- Em linhas gerais, é indicada a criação de índice nas colunas utilizadas nas seguintes cláusulas da Linguagem SQL:

- |            |            |
|------------|------------|
| ✓ WHERE    | ✓ GROUP BY |
| ✓ ORDER BY | ✓ HAVING   |

# Tipos de Índices

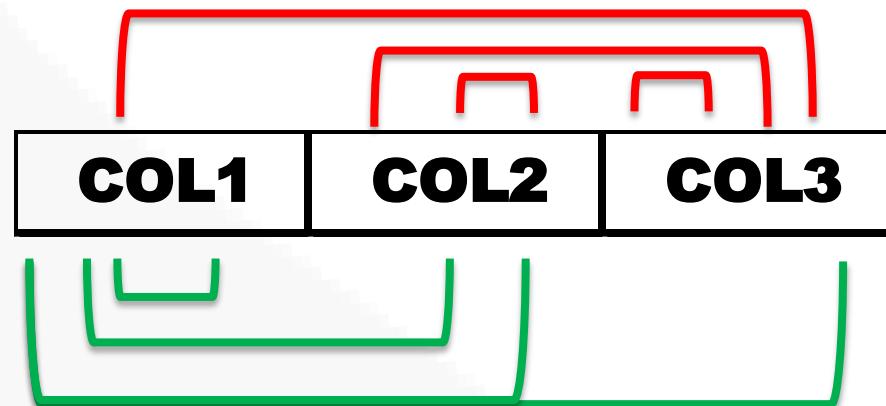
- Em termos de **composição**, o índice pode ser:
  - **SIMPLES** (composto por apenas uma coluna);
  - **COMPOSTO** (composto por duas ou mais colunas).
  - **Exemplo:** cláusula WHERE utilizando-se as colunas COL1 e COL2.
    - Índice simples para a coluna COL1 e outro índice simples para a coluna COL2 → **não atende**.
    - Índice composto pelas colunas COL1 e COL2 → **será usado pela consulta**.

# Tipos de Índices

- Em termos de **funcionalidade**, cada fornecedor disponibiliza os tipos de índices mais aderentes ao seu SGBD.
- **ÍNDICE ÚNICO**: não permite valores repetidos para a(s) coluna(s).
- **ÍNDICE NÃO ÚNICO**: permite valores repetidos para a(s) coluna(s).
- **ÍNDICE PRIMÁRIO**: sempre único, correspondendo à chave primária da tabela.
- **ÍNDICE SECUNDÁRIO**: pode ser único (chave candidata) ou não único.

# Shift Index

- Índices compostos que conseguem ser usados por uma query, mesmo ela não usando todas as colunas que compõem o índice.
  - Primeira coluna do índice precisa estar sendo usada na query;
  - Em alguns SGBDs, as colunas usadas na query devem estar na sequência que foram criadas no índice.



# Efeitos Colaterais dos Índices



- Índices ocupam **espaço adicional**;
- Índices podem tornar as **operações de carga, exclusão ou atualização de dados** mais lentas, devido às possíveis operações de reordenação e/ou atualização do índice;
- Índices podem interferir na **cardinalidade** dos relacionamentos existentes no modelo de dados físico;
- Índices precisam de **manutenção periódica** para remoção da fragmentação → **rebuild / reorg** e **atualização de estatísticas**.

# Próxima Aula



- Estatísticas e Plano de Execução de Query.

# Aspectos Gerais de Performance em Banco de Dados

---

CAPÍTULO 5. AULA 5.4. ESTATÍSTICAS E PLANO DE EXECUÇÃO DE QUERY

PROF. GUSTAVO AGUILAR

# Nesta Aula



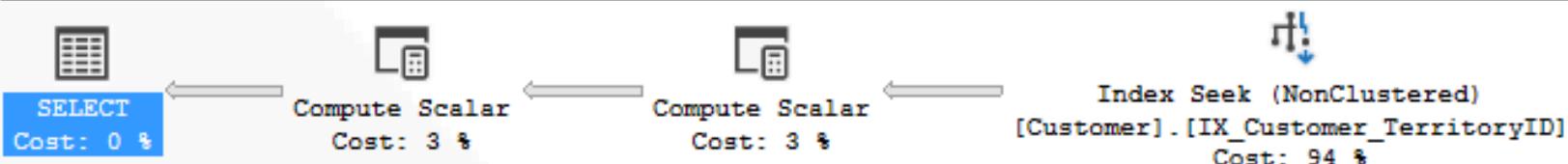
- Plano de Execução de Queries.
- Estatísticas.

# Plano de Execução de Queries



- Usado pelo SGBD para indicar como cada query dever ser executada;
- Contém os **objetos que serão utilizados na query**, os índices e mecanismos usados.

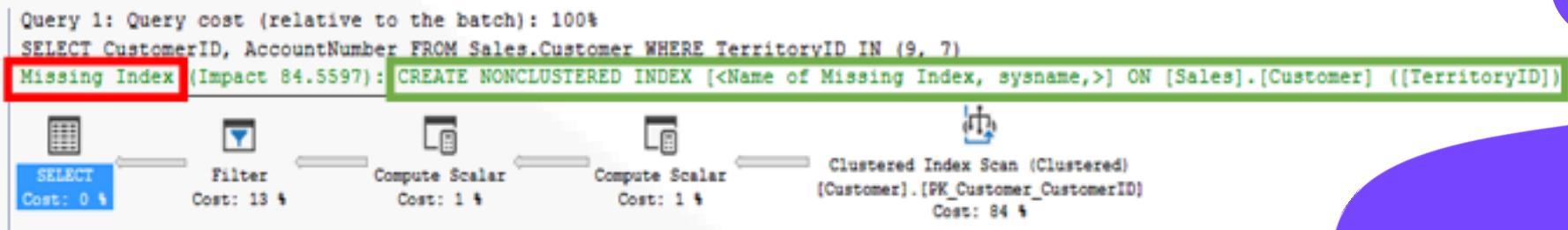
```
SELECT CustomerID, AccountNumber FROM Sales.Customer WHERE TerritoryID IN (9, 7)
```



# Plano de Execução de Queries



- **Análise do Plano de Execução da Query:** etapa fundamental e que reduzirá as chances de execução de queries que gerarão impactos negativos na performance do ambiente;
- Grande maioria dos SGBDS disponibiliza interfaces gráficas para visualizar o plano de execução, e algumas até sugerem a criação de índices, inclusive já com o comando DDL para tal.



# Estatísticas

- Uma amostra dos dados em uma determinada tabela / coluna de índice.

Statistics for Sales.SalesOrderDetail (estimated)							
Name	Updated	Rows	Rows Sampled	Steps	Density	Average key length	String Index
IX_SalesOrderDetail_ProductID	Nov 7 2012 6:44PM	121317	121317	200	0.0078125	12	NO
RANGE_HI_KEY	RANGE_ROWS	EQ_ROWS	DISTINCT_RANGE_ROWS	AVG_RANGE_ROWS			
730	0	288	0	1			
732	0	130	0	1			
738	154	600	2	77			
741	167	94	1	167			
742	0	288	0	1			

```
SELECT
    ProductID, RecordCount = COUNT(*)
FROM Sales.SalesOrderDetail
WHERE
    ProductID >= 732 AND
    ProductID <= 738
GROUP BY ProductID
```

Statistics for Sales.SalesOrderDetail (estimated)	
ProductID	RecordCount
732	130
733	44
736	110
738	600

# Estatísticas

- O otimizador usa esses valores para determinar o melhor método de acesso para determinadas operações de dados;
- Valores desatualizados ou distorcidos podem levar a planos de execução ineficientes;
  - Requerem manutenção constante.
- Podem ser criadas / atualizadas manualmente ou automaticamente.

# Próxima Aula



- Capítulo 6 - Otimização de Query no SQL Server.

# Performance e Otimização

Capítulo 6. Otimização de Query no SQL Server

PROF. GUSTAVO AGUILAR

# Otimização de Query no SQL Server

---

CAPÍTULO 6. AULA 6.1. TIPOS DE ÍNDICES

PROF. GUSTAVO AGUILAR

# Nesta Aula



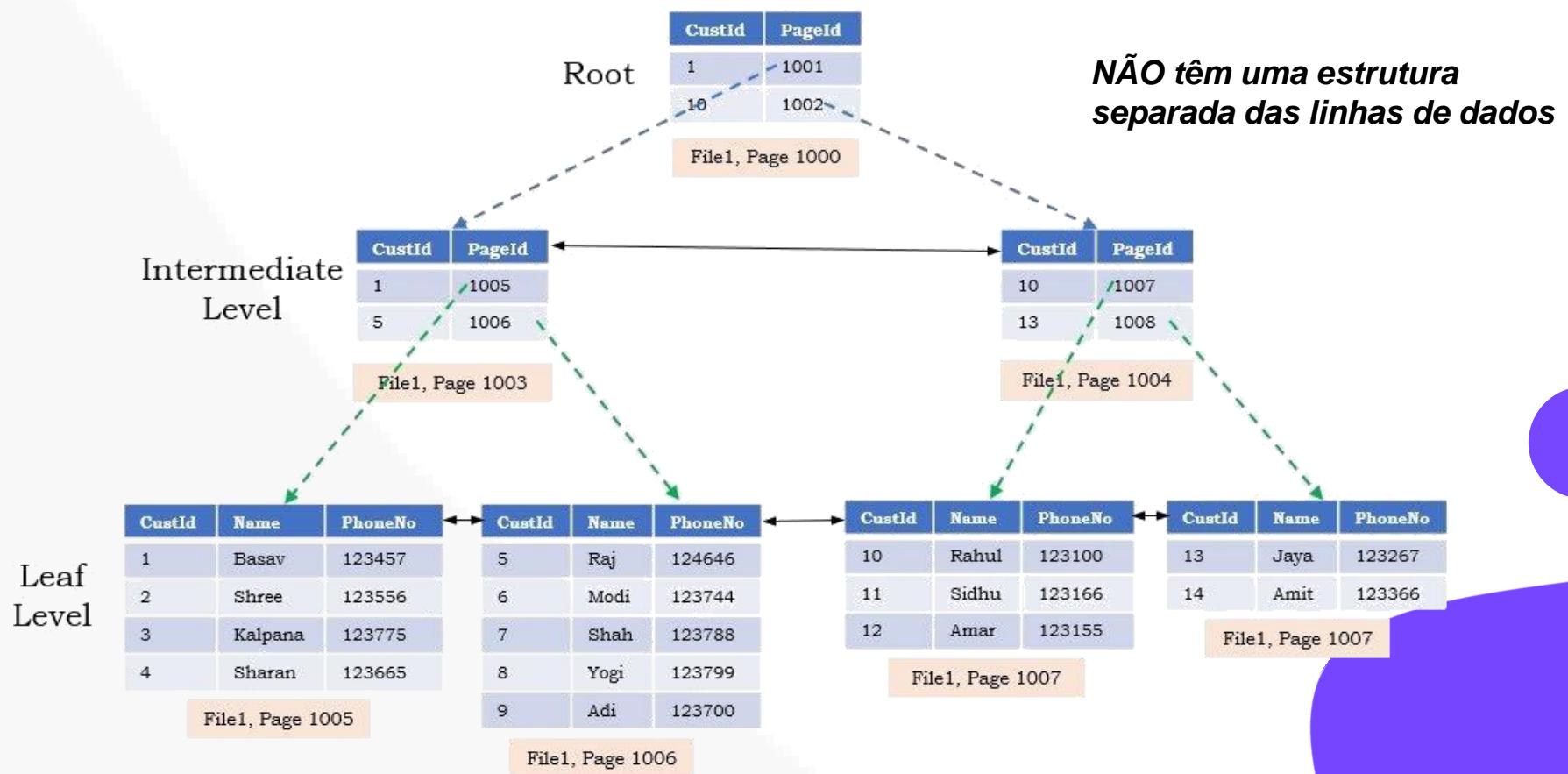
- Clustered Index.
- Nonclustered Index.
- Heaps.
- Full Text Index.
- Columnstore Index.
- Outros Tipos de Índices.

# Clustered Index

- Índice clusterizado / agrupado → podem ser únicos ou não únicos;
- **Classificam e armazenam as linhas na tabela ou view** com base em seus valores-chave (colunas incluídas na criação do índice);
- Apenas um índice clusterizado por tabela;
  - As linhas na tabela só podem ser armazenadas em uma ordem;
- Tabela é chamada de **tabela clusterizada (*clustered table*)**.

# Clustered Index

B+ Tree Structure of a Clustered Index



# Clustered Index

- Criado com o comando:

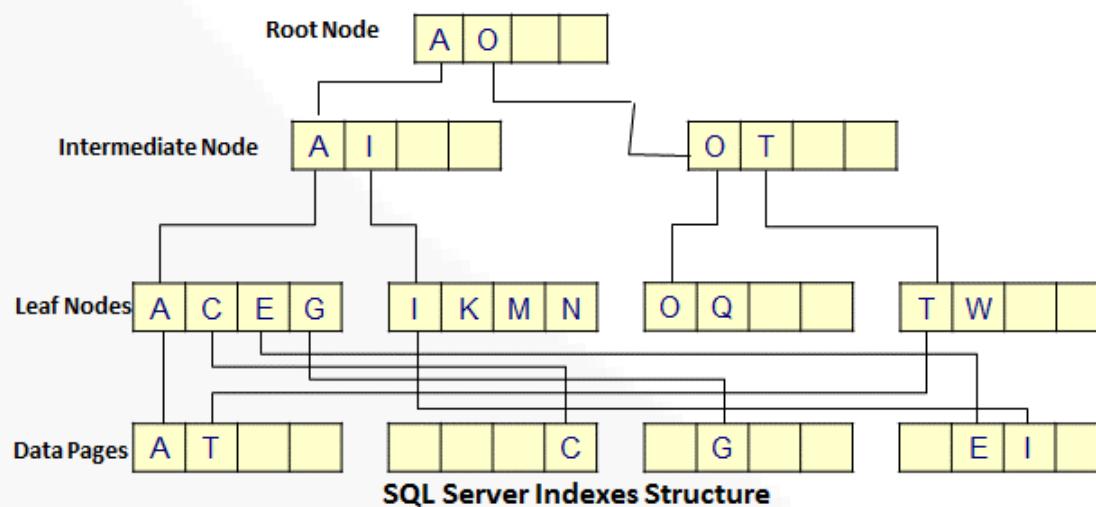
```
CREATE [UNIQUE] CLUSTERED INDEX NOME_DO_INDICE  
ON NOME_DA_TABELA (COLUNA1, [COLUNA2],...)
```

- Criado automaticamente quando se define uma constraint primary key:

```
CREATE TABLE NOME_DA_TABELA  
(      COLUNA1,  
          COLUNA2....  
          PRIMARY KEY CLUSTERED  
          (COLUNA1)  
          ON [PRIMARY]  
      ) ON [PRIMARY]
```

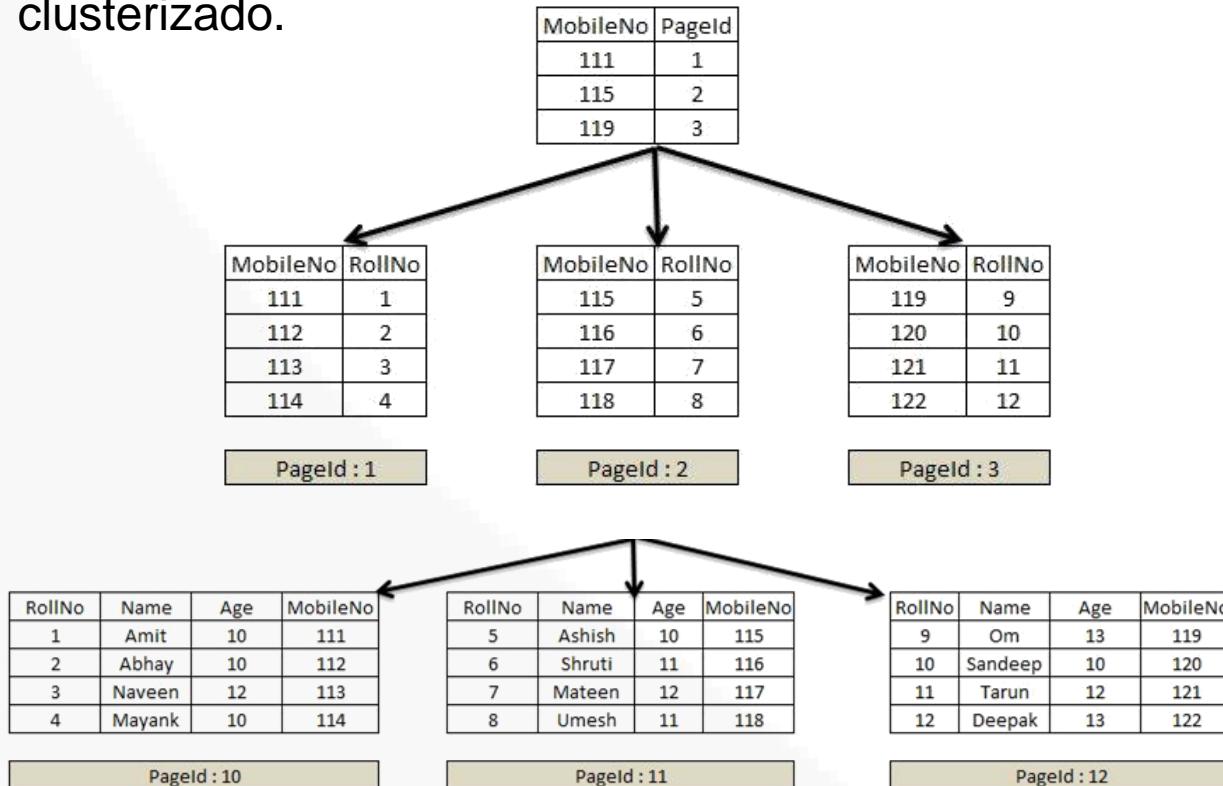
# Non Clustered Index

- Índices não clusterizados → podem ser únicos ou não únicos;
- **Têm** uma estrutura separada das linhas de dados da tabela;
- Um índice não clusterizado contém os valores-chave do índice não clusterizado e cada entrada de valor-chave tem um ponteiro para a linha de dados que contém esse valor-chave.



# Non Clustered Index

- Quando há um índice cluster na tabela, a página de nível folha do índice não clusterizado (leaf node) também inclui a chave do índice clusterizado.



# Non Clustered Index

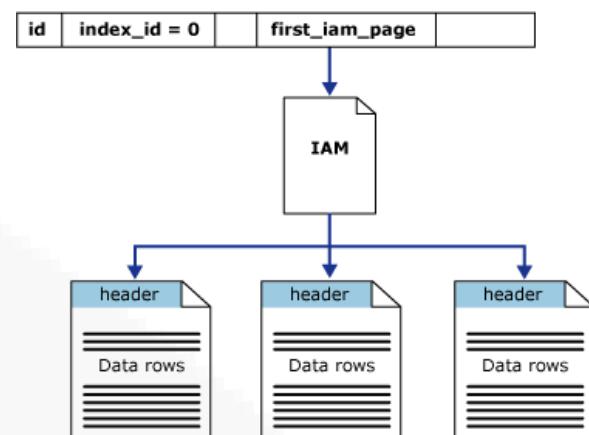
- Criado com o comando:

```
CREATE NONCLUSTERED INDEX NOME_DO_INDICE  
ON NOME_DA_TABELA (COLUNA1, [COLUNA2],...)
```

```
CREATE UNIQUE NONCLUSTERED INDEX NOME_DO_INDICE  
ON NOME_DA_TABELA (COLUNA1, [COLUNA2],...)
```

# Heap

- Tabela sem índice clusterizado;
- Os dados não são armazenados em uma ordem específica;
- Dados específicos não podem ser recuperados rapidamente, a menos que também exista índice não clusterizado na tabela;
- As páginas de dados não estão vinculadas, portanto, o acesso sequencial precisa consultar as páginas do mapa de alocação de índice (IAM).



# Heap

- Linhas são identificadas por um ID de 8 bytes (RID):
  - Número do arquivo, da página de dados e do slot (FileID: PageID: SlotID).
- Necessário ler a tabela inteira (full table scan) para localizar um dado;
- Pode possuir um ou mais índices **não** clusterizados;
- Quando não há índice clusterizado, não há necessidade de espaço adicional para armazenar a árvore de índice clusterizado, e também não é necessário tempo adicional para manter o índice ordenado.
- **Quando usar:**
  - ✓ Como tabelas temporárias para operações de inserção não ordenadas;
  - ✓ Quando os dados são sempre acessados por meio de índices não clusterizados ou o RID é menor que uma chave de índice clusterizado.

# Heap

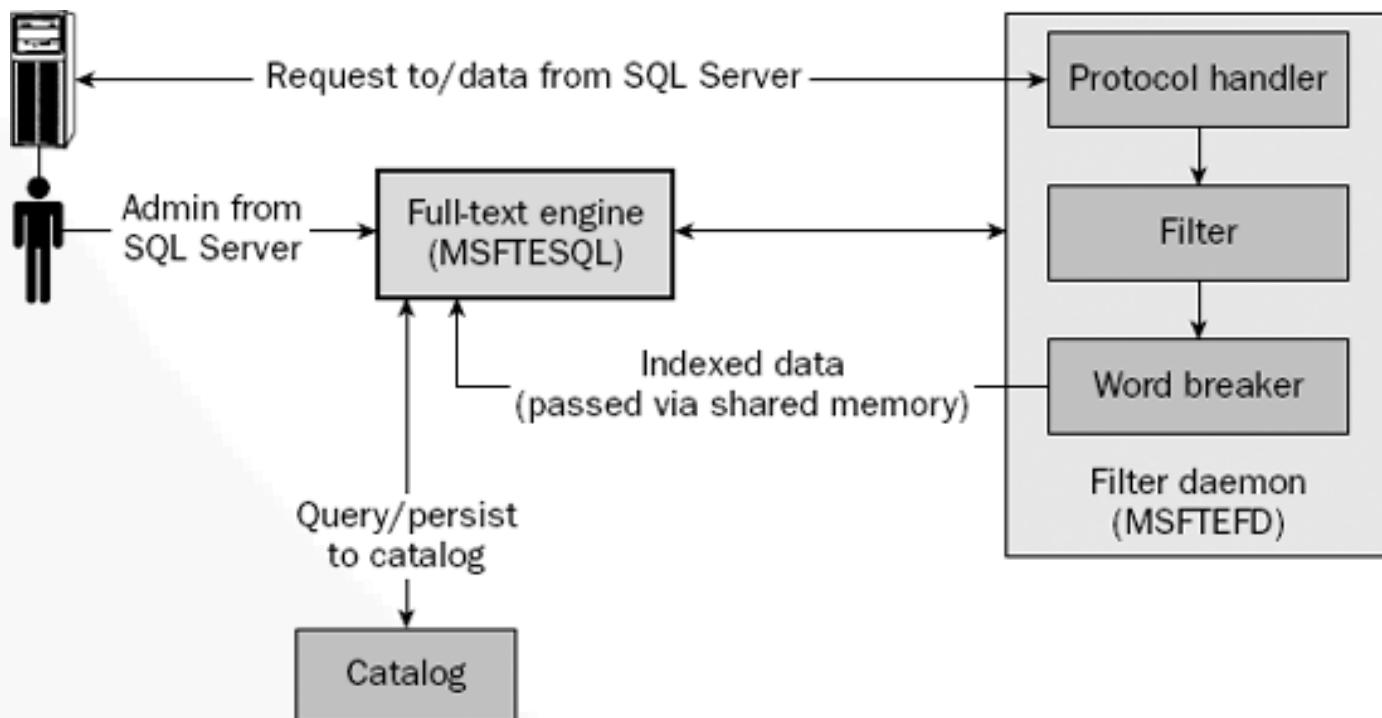
## ▪ Quando NÃO usar:

- ✓ Quando os dados são freqüentemente retornados em uma ordem de classificação → use índice clusterizado na coluna de classificação;
- ✓ Quando os dados forem frequentemente agrupados (os dados devem ser classificados antes de serem agrupados) → um índice clusterizado na coluna de classificação pode evitar a operação de classificação (sort).
- ✓ Quando intervalos de dados são consultados com frequência na tabela → um índice clusterizado na coluna de intervalo evitaria a classificação da heap toda.
- ✓ Quando não houver índices não clusterizados e a tabela for grande, a menos que você pretenda retornar todo o conteúdo da tabela sem qualquer ordem especificada;
- ✓ Não use um heap se os dados forem atualizados com frequência.

# Full Text Index

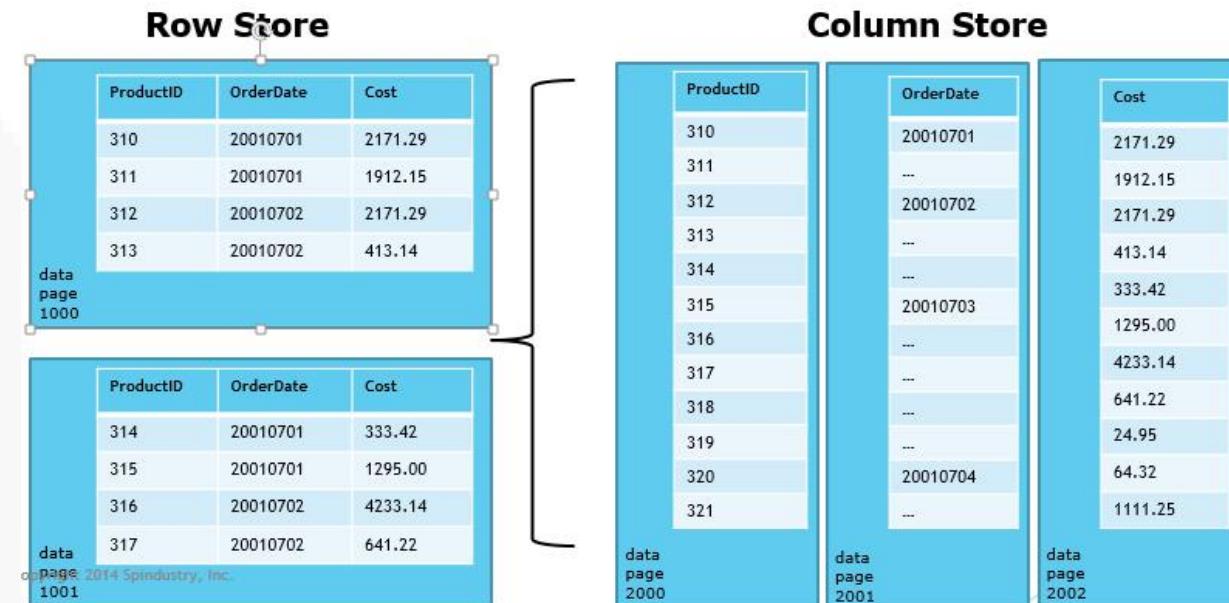
- Usado para filtros complexos em string de dados;
  - Permite consultas semânticas nos dados;
  - Pesquisas em campos binários (PDF), etc.
- Um *full text index* por tabela;
- Requer uma coluna não nula única na tabela;
- Requer a instalação da engine (serviço) Full-Text;
- Necessário criar e manter os catálogos;
- Comandos T-SQL específicos Specialized Transact-SQL (for example:  
*WHERE CONTAINS(Name, 'formsof(freetext, nut)').*

# Full Text Index



# Columnstore Index

- Recurso para armazenar, recuperar e gerenciar dados usando um formato de dados **colunar**, chamado **columnstore**;
- Um columnstore são dados logicamente organizados como uma tabela com linhas e colunas, e fisicamente armazenados em formato colunar de dados.



# Columnstore Index

- As colunas armazenam valores do mesmo domínio e geralmente têm valores semelhantes, o que resulta em altas taxas de compactação.
  - Altas taxas de compactação melhoram o desempenho da consulta usando um espaço menor na memória.
- **Quando usar:**
  - Para armazenar tabelas de fatos e tabelas grandes de dimensão;
  - Para realizar análises em tempo real em uma carga de trabalho OLTP.

# Outros Tipos de Índices

## ■ XML Index

	FirstName	LastName	Address
1	Desiree	Moyer	<Address state="MI" zip="09219"><Street>63 First S...
2	Desiree	Ware	<Address state="MI" zip="97221"><Street>29 Nobel ...
3	Desiree	Andrade	<Address state="MI" zip="03064"><Street>941 Roc...
4	Desiree	Chung	<Address state="MI" zip="44802"><Street>59 White ...
5	Desiree	Wilkerson	<Address state="MI" zip="63385"><Street>30 White ...
6	Desiree	Davidson	<Address state="MI" zip="82593"><Street>84 West ...
7	Desiree	Ward	<Address state="MI" zip="27369"><Street>735 Milton ...
8	Desiree	Floyd	<Address state="MI" zip="19197"><Street>245 Rock ...

```
<Employee ssn="628599159" dob="1972-01-27">
  <FirstName>Nicole</FirstName>
  <MiddleName>Angie</MiddleName>
  <LastName>Fuller</LastName>
  <Address state="MI" zip="06837">
    <Street>131 Milton Street</Street>
    <City>Glendale</City>
  </Address>
  <Internal ID="1">
    <Salary>70284</Salary>
    <Hiredate>2006-07-29</Hiredate>
    <Department>Corporate Care</Department>
  </Internal>
</Employee>
```

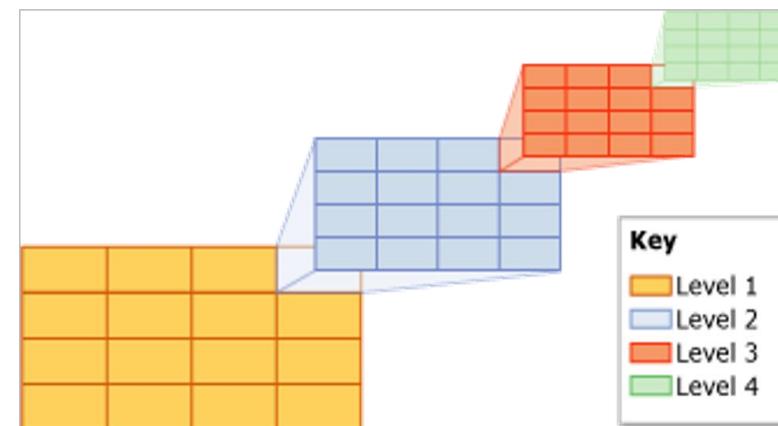
# Outros Tipos de Índices

## ▪ Spatial Index (Índice Espacial)

- Para tipos de dados de geografia (*geography*) e geometria (*geometry*);
- O índice é criado como uma hierarquia com quatro níveis.

```
CREATE TABLE SpatialTable  
(id int primary key, geometry_col geometry);
```

```
CREATE SPATIAL INDEX SIdx_SpatialTable_geometry_col1  
ON SpatialTable(geometry_col)  
WITH ( BOUNDING_BOX = ( 0, 0, 500, 200 ) );
```



# Próxima Aula



- Demonstração: Criando Índices.

# Otimização de Query no SQL Server

---

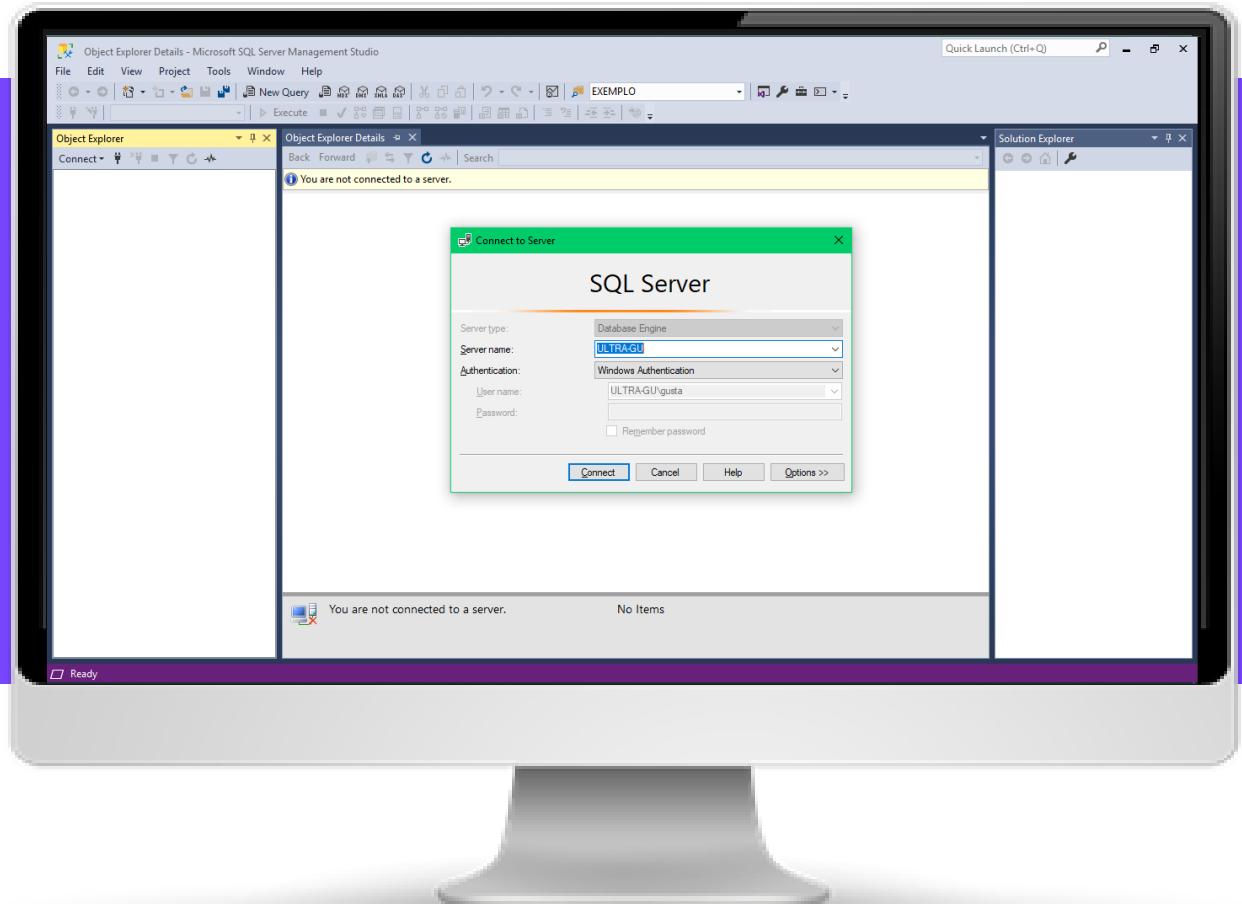
CAPÍTULO 6. AULA 6.2. DEMONSTRAÇÃO: CRIANDO ÍNDICES

PROF. GUSTAVO AGUILAR

# Criando Índices



Demo



# Próxima Aula



- Análise do Plano de Execução de Query.

# Otimização de Query no SQL Server

---

CAPÍTULO 6. AULA 6.3. ANÁLISE DO PLANO DE EXECUÇÃO DE QUERY

PROF. GUSTAVO AGUILAR

# Nesta Aula



- ❑ Conteúdo do Plano de Execução de Query.
- ❑ Análise do Plano de Execução de Query.
- ❑ Operadores.
- ❑ Dicas para Otimizar Planos de Query.

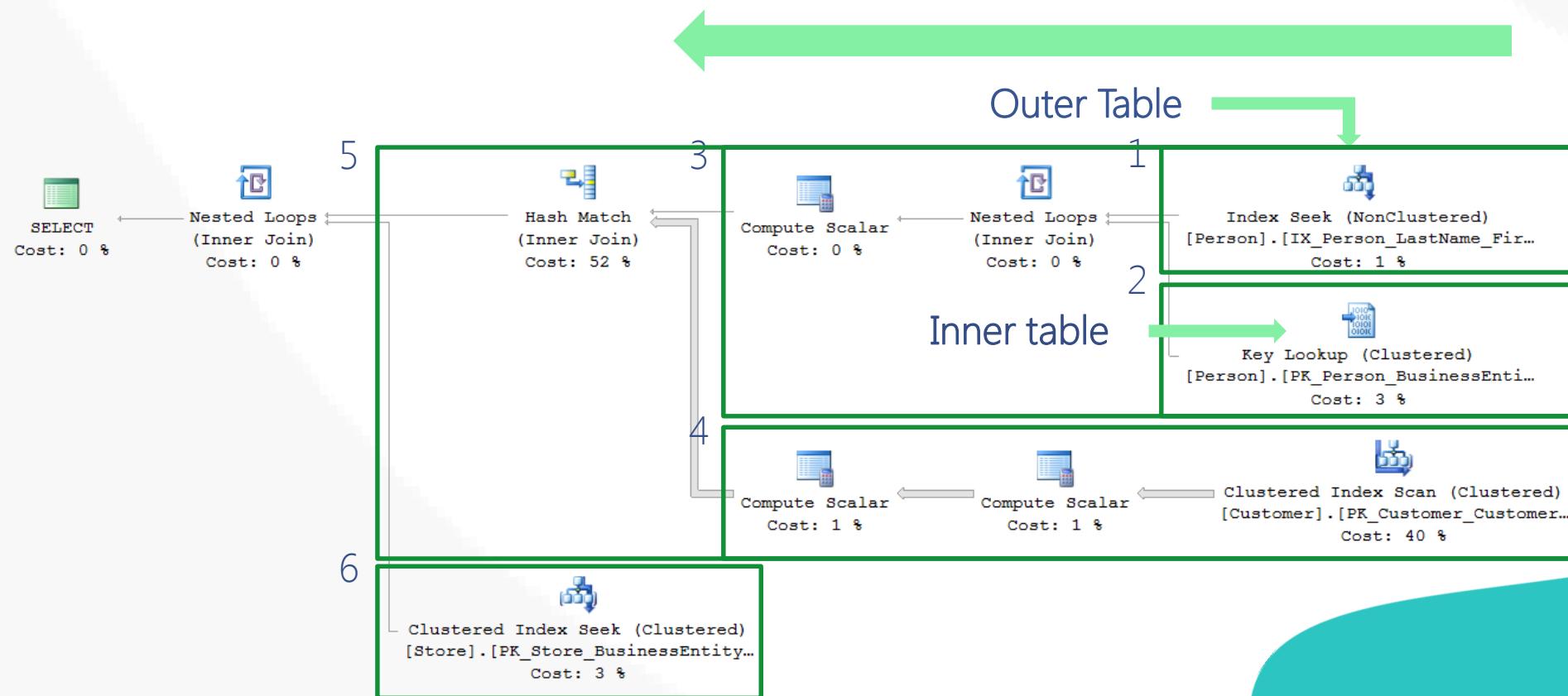
# Conteúdo do Plano de Execução



- ✓ Como os dados são acessados;
- ✓ Como os dados são unidos;
- ✓ Sequência de operações;
- ✓ Uso de tabelas de trabalho temporárias e classificações;
- ✓ Contagens de linhas estimadas, iterações e custos de cada etapa;
- ✓ Contagens de linhas reais e iterações;
- ✓ Como os dados são agregados;
- ✓ Uso de paralelismo;
- ✓ Warnings de execução da query.

# Análise do Plano de Execução

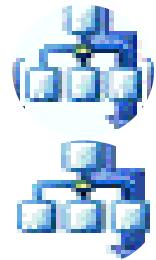
IGTI



# Operadores

## ▪ Operadores Seek

✓ *Index Seek*



✓ *Clustered Index Seek*



- O operador *Clustered Index Seek* usa a capacidade dos índices para recuperar linhas de um índice clusterizado (clustered index);
- O operador *Index Seek* usa a capacidade de busca de índices para recuperar linhas de um índice não clusterizado.

# Operadores

## ▪ Operadores Lookup

✓ *Row Identifier (RID) Lookup*



✓ *Key Lookup*

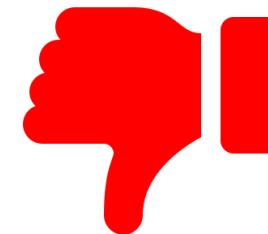


- *RID Lookup* é uma pesquisa em uma tabela **HEAP** usando um identificador de linha fornecido (RID) para pesquisar a linha na tabela, e o nome da tabela na qual a linha é pesquisada.
- *Key Lookup* é uma pesquisa em uma tabela com um índice clusterizado. É sempre acompanhada por um operador de loops aninhados. Esse operador pode indicar que a consulta pode se beneficiar se for criado um índice mais abrangente.

# Operadores

## ▪ Operadores Scan

- ✓ *Table Scan*
- ✓ *Clustered Index Scan*
- ✓ *Index Scan*
- ✓ *Columnstore Index Scan*



- O operador *Table scan* percorre todas as linhas da tabela para retornar as linhas desejadas na query.
- Demais operadores percorrem toda a estrutura do índice para retornar as linhas desejadas na query.

# Operadores

## ▪ Outros operadores

✓ *Table Spool*



✓ *Index Spool*



✓ *Sort*



- *Table Spool* coloca uma cópia de cada linha em uma tabela de spool oculta que é armazenada no banco de dados *tempdb* e existe apenas durante o tempo de vida da query.
- *Index Spool* coloca uma cópia de cada linha em um arquivo de spool oculto (armazenado no *tempdb* e existindo apenas durante o tempo de vida da consulta) e cria um índice não cluster → permite usar a capacidade de busca de índices (*Index Seek*).

# Dicas para Otimizar Plano de Query



## Revisar e otimizar os JOINS

- Nested loops apenas para conjuntos pequenos de dados.

## Rever operadores SCANS e RID

- Removê-los usando índices que contemplam toda a query.

## Rever operadores Key Lookups

- Removê-los usando índices que contemplam toda a query.

## Outras considerações

- Eliminar spools.

# Dicas para Otimizar Plano de Query



Limitar a quantidade de JOINS

Limitar a quantidade de linhas nos JOINS

Criar índices nas colunas dos JOINS

Fazer JOINS usando as colunas mais únicas

Fazer JOINS com colunas do mesmo tipo de dados (evitar conversão implícita)

Evitar usar SELECT \*

- Sempre selecione explicitamente as colunas que você deseja retornar.

Usar a sintaxe ANSI nos JOINS

- `SELECT fname, lname, department  
FROM names INNER JOIN departments  
ON names.employeeid = departments.employeeid`

# Dicas para Otimizar Plano de Query



Evitar lógicas negativas, como !=, <>, NOT (...)

- Isso introduz contenção adicional, porque muitas vezes resulta na avaliação de cada linha (index scan).

Só usar ORDER BY se for estritamente necessário

- Se estiver coberto por um índice, garanta que a ordenação do índice é a mesma da clausula ORDER BY.

Evitar o operador LIKE com caracteres curingas ('%VALOR%'), pois sempre causará um table scan

- Se tiver que usar LIKE, torne o primeiro caracter um literal ('VALOR%').

# Próxima Aula



- Demonstração: Análise do Plano de Execução de Query.

# Otimização de Query no SQL Server

---

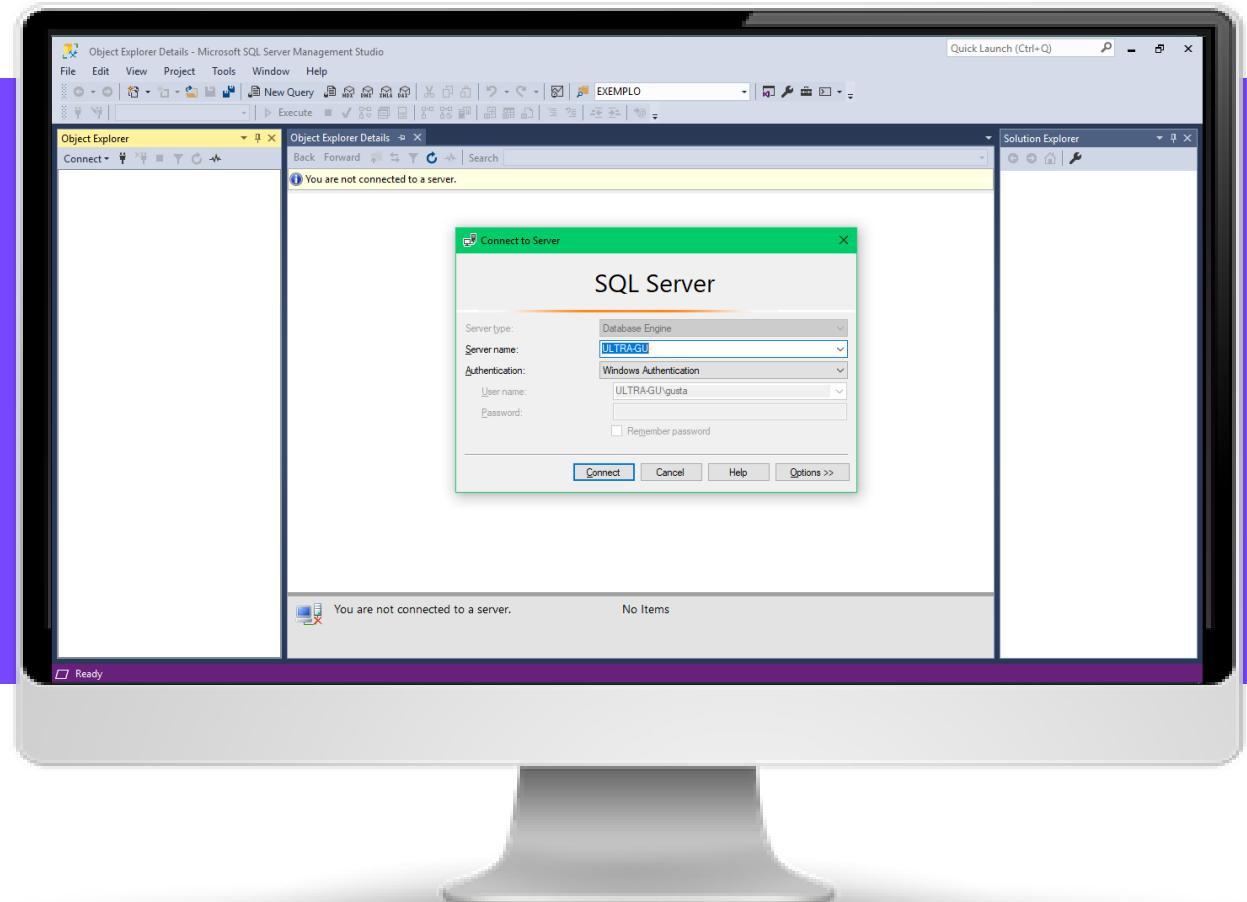
CAPÍTULO 6. AULA 6.4. DEMONSTRAÇÃO: ANÁLISE DO PLANO DE EXECUÇÃO DE QUERY

PROF. GUSTAVO AGUILAR

# Análise do Plano de Execução de Queries



 Demo



# Próxima Aula



- ❑ Activity Monitor e Performance Dashboard.

# Otimização de Query no SQL Server

---

CAPÍTULO 6. AULA 6.5. ACTIVITY MONITOR E PERFORMANCE DASHBOARD

PROF. GUSTAVO AGUILAR

# Nesta Aula



- Activity Monitor.
- Performance Dashboard.

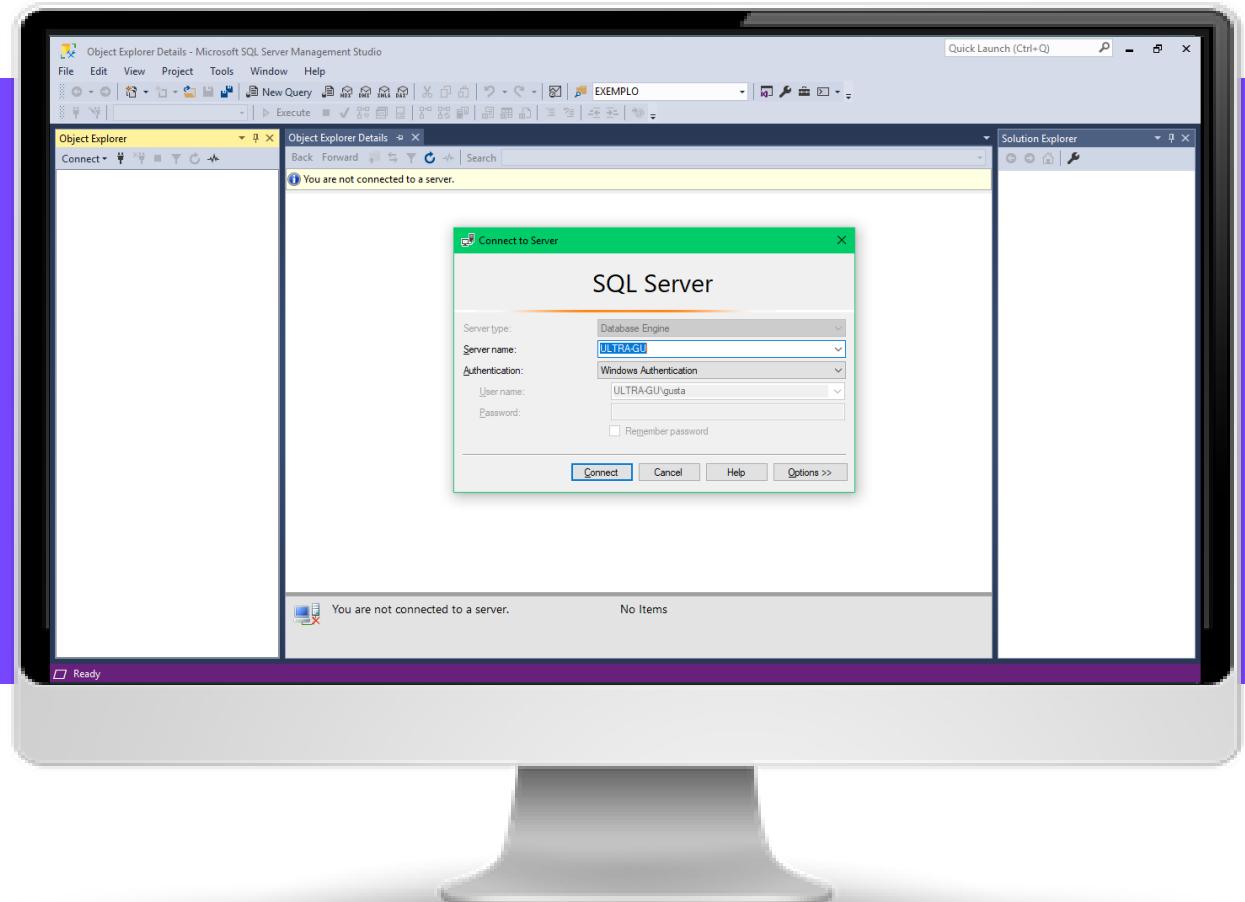
# Activity Monitor



## Performance Dashboard



# Demo



# Próxima Aula



- SQL Trace e Extended Events.

# Otimização de Query no SQL Server

---

CAPÍTULO 6. AULA 6.6. SQL TRACE E EXTENDED EVENTS

PROF. GUSTAVO AGUILAR

# Nesta Aula



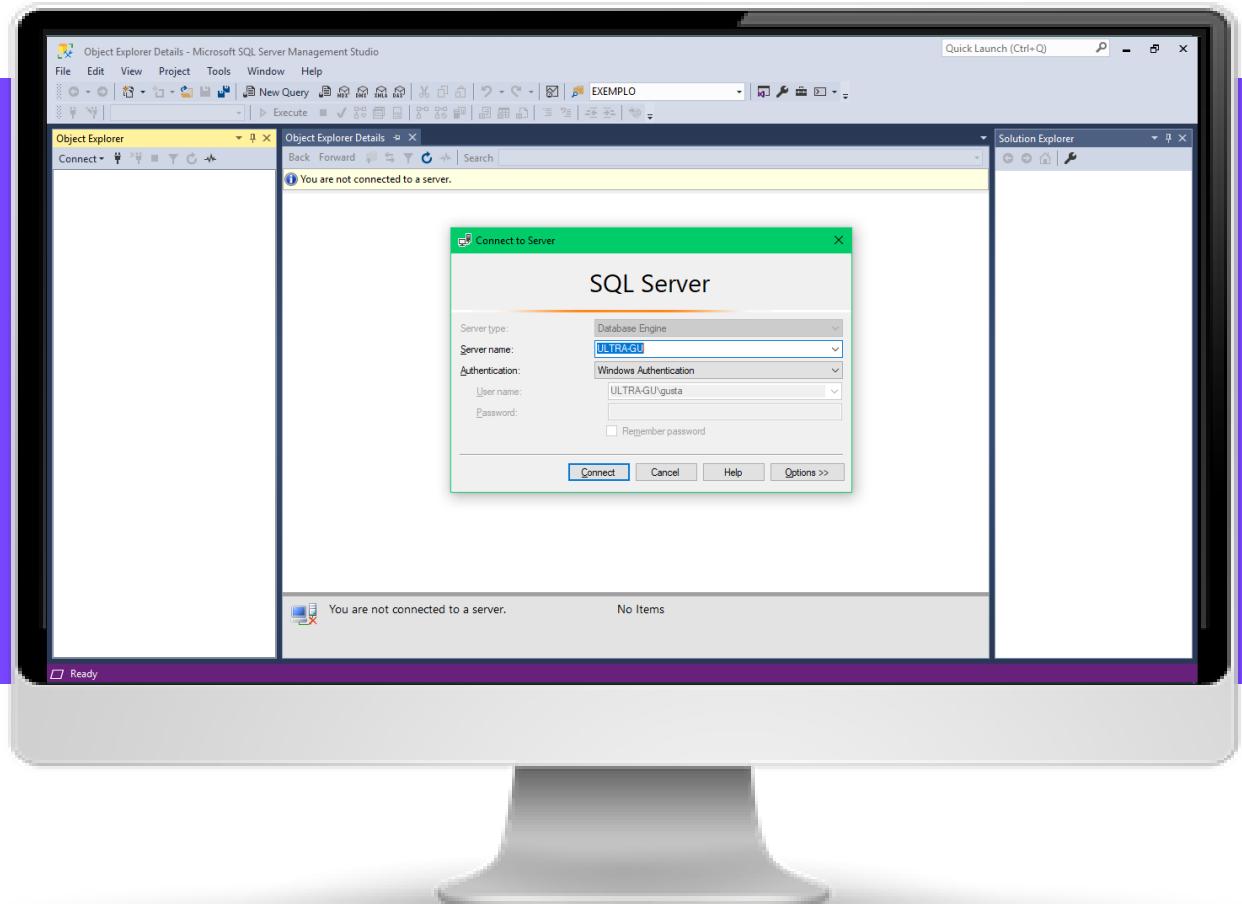
- SQL Trace.
- Extended Events.

# SQL Trace



# Extended Events

Demo



# Próxima Aula



- Query Store e Database Tuning Advisor (DTA).

# Otimização de Query no SQL Server

---

CAPÍTULO 6. AULA 6.7. QUERY STORE E DATABASE TUNING ADVISOR (DTA)

PROF. GUSTAVO AGUILAR

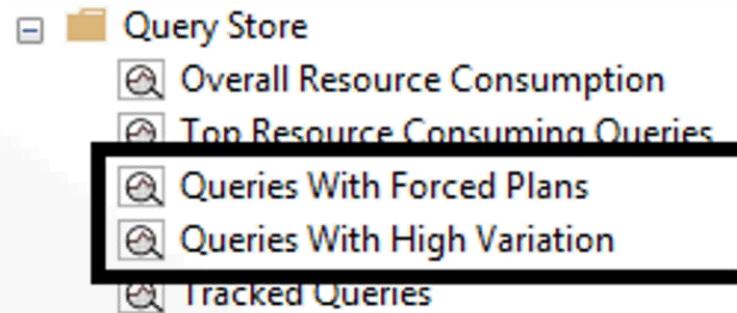
# Nesta Aula



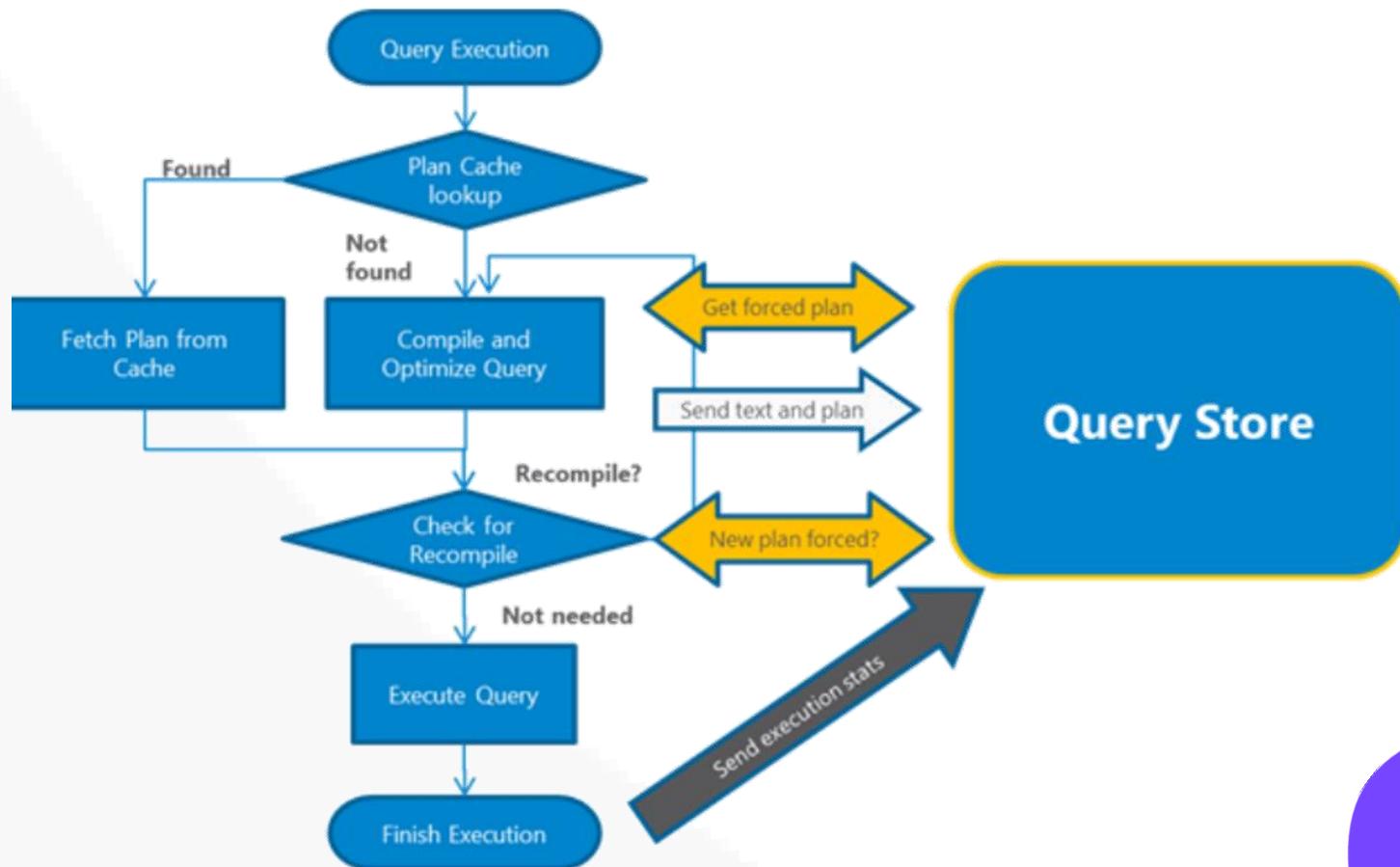
- Query Store.
- Database Tuning Advisor (DTA).

# Query Store

- ✓ Armazena o histórico de planos para cada query;
- ✓ Cria uma linha de base para o desempenho de cada plano ao longo do tempo;
- ✓ Identifica as consultas que ficaram mais lentas recentemente;
- ✓ É possível forçar planos de forma rápida e fácil;
- ✓ Trabalha ao longo das reinicializações de servidor, atualizações e recompilação de queries.

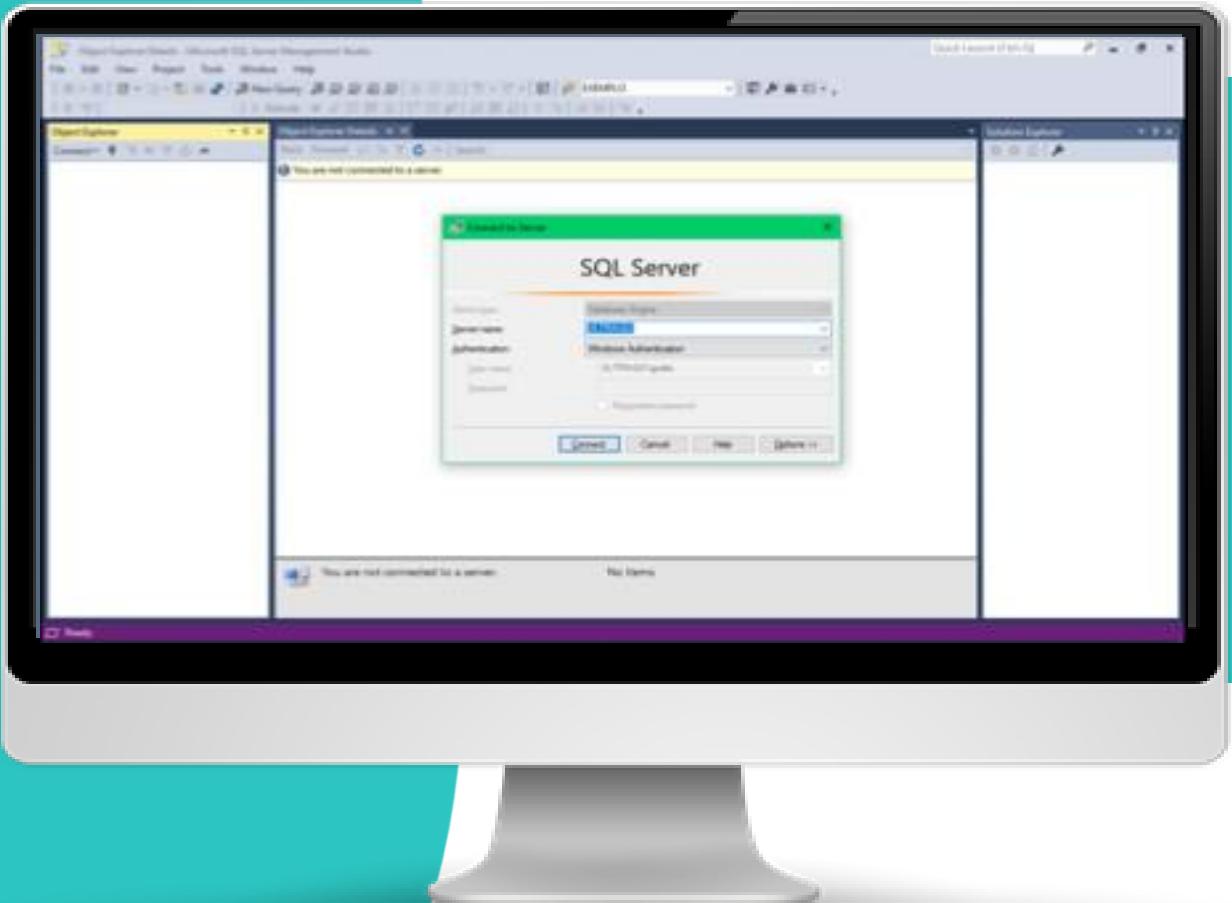


# Query Store



# Database Tuning Advisor (DTA)

Demo



# Próxima Aula



- Memory-Optimized Tables.

# Otimização de Query no SQL Server

---

CAPÍTULO 6. AULA 6.8. MEMORY-OPTIMIZED TABLES

PROF. GUSTAVO AGUILAR

# Nesta Aula



- Memory-Optimized Tables.
- Transaction Performance Analysis Report.
- Memory Optimization Advisor.
- Hash Index.

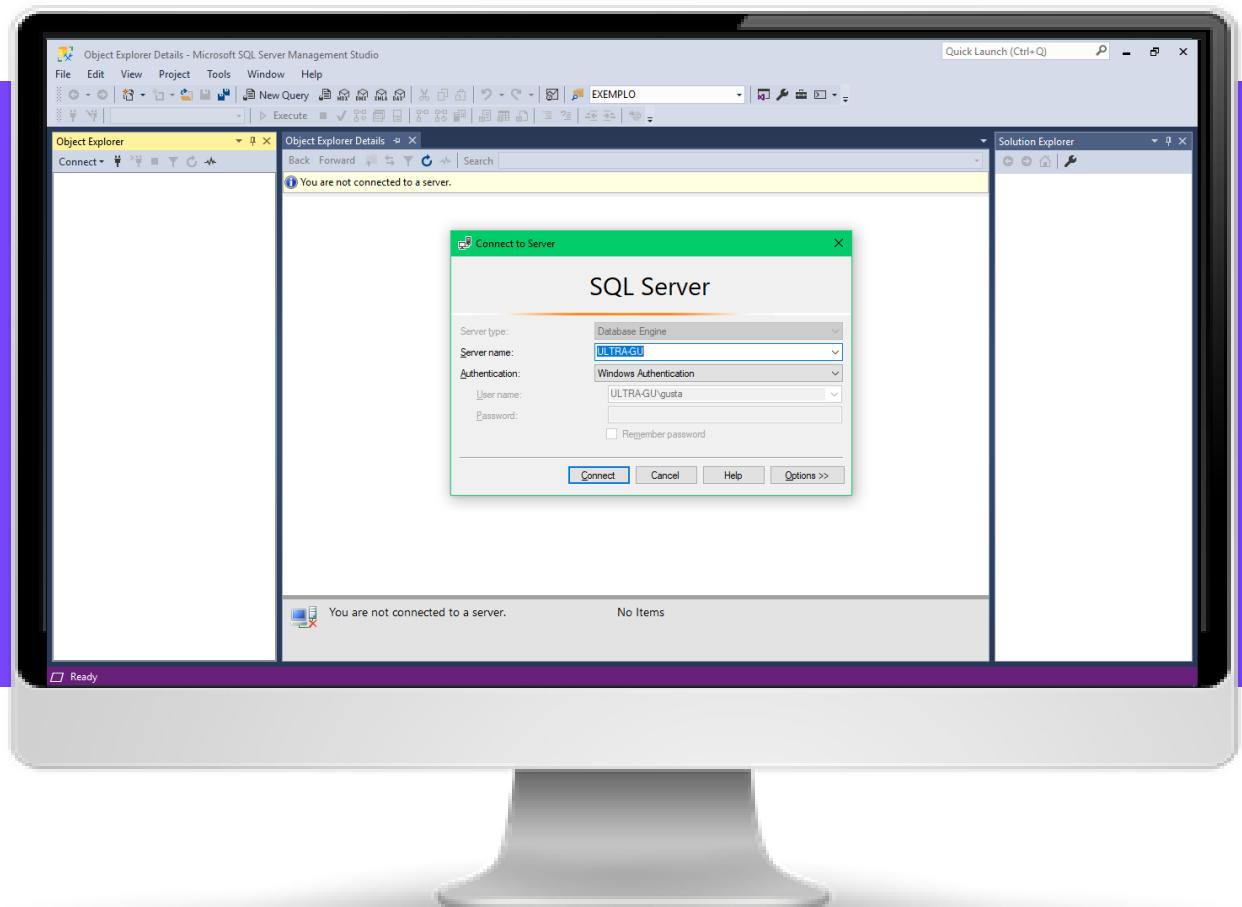
# Memory-Optimized Tables

- Tabela cujo meio de armazenamento primário é a memória principal (**RAM**);
  - As linhas da tabela são lidas e gravadas na memória;
- As tabelas *memory-optimized* são totalmente **duráveis por padrão**;
  - Uma segunda cópia dos dados da tabela é mantida em disco, mas apenas para fins de durabilidade;
  - Os dados em tabelas *memory-optimized* só são lidos do disco durante a recuperação do banco de dados (por ex., após a reinicialização do servidor).
- Transações em tabelas *memory-optimized* são totalmente atômicas, consistentes, isoladas e duráveis (**ACID**).
- Além das tabelas *memory-optimized* duráveis por padrão, o SQL Server também oferece suporte a tabelas *memory-optimized* **não duráveis**.

# Memory-Optimized Tables



 Demo

The text "Demo" is displayed on a purple background. To its left is a white play button icon.

# Próxima Aula



- ❑ Capítulo 7 - Otimização de Query no MongoDB.

# Performance e Otimização

Capítulo 7. Otimização de Query no MongoDB

PROF. GUSTAVO AGUILAR

# Otimização de Query no MongoDB

---

CAPÍTULO 7. AULA 7.1. TIPOS DE ÍNDICES

PROF. GUSTAVO AGUILAR

# Nesta Aula



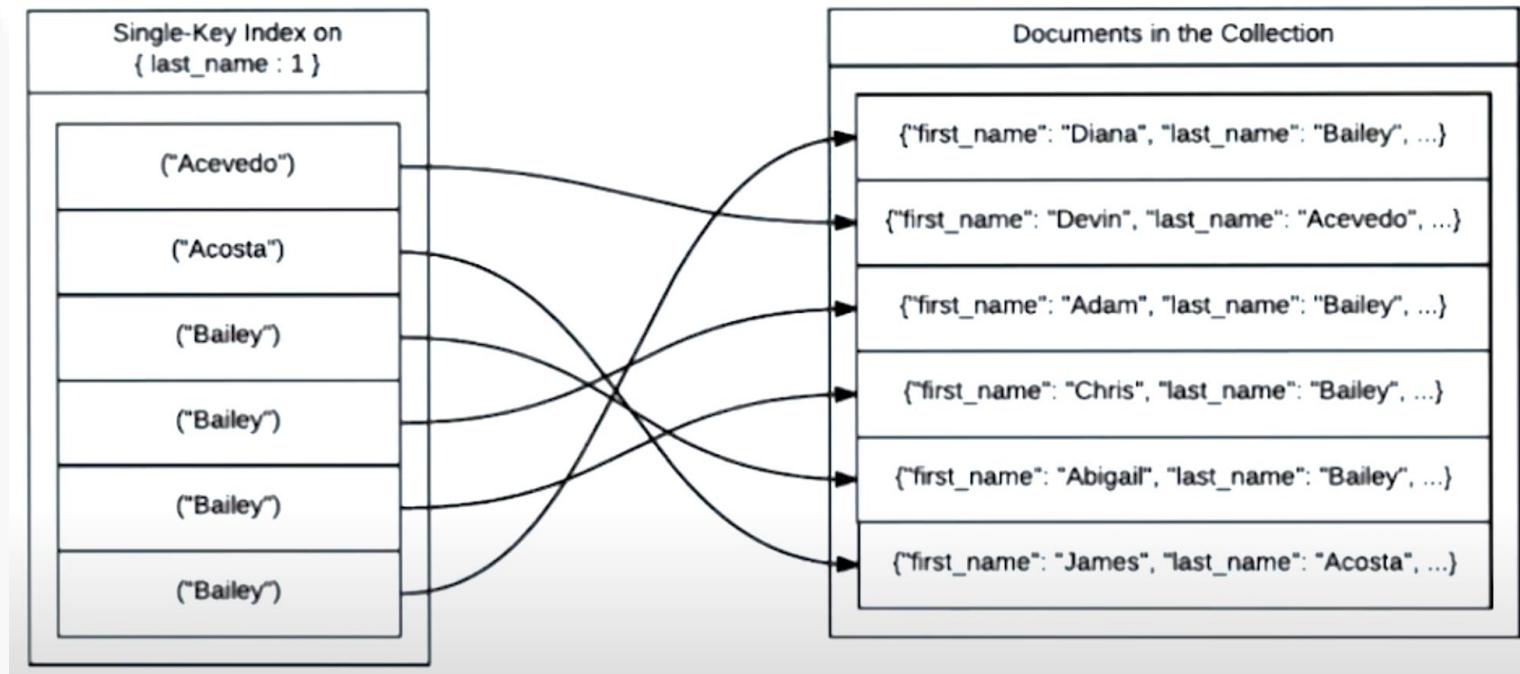
- Default Index.
- Single Field Index.
- Compound Index.
- Multikey Index.
- Text Index.
- Partial Index.
- Covered Query.
- Index Intersection.

# Índices

- Os índices são **estruturas de dados especiais** que armazenam uma pequena parte do conjunto de dados da collection, de uma forma fácil de se percorrer;
  - Se existir um índice apropriado para uma query, o MongoDB pode usar o índice para **limitar o número de documentos que deve inspecionar**.
- O índice armazena o valor de um campo específico ou conjunto de campos, **ordenado pelo valor do campo**;
- Oferece suporte a **comparações de igualdade** e operações de consulta **baseadas em intervalo**;
- O MongoDB pode **retornar resultados classificados** usando a ordenação no índice.

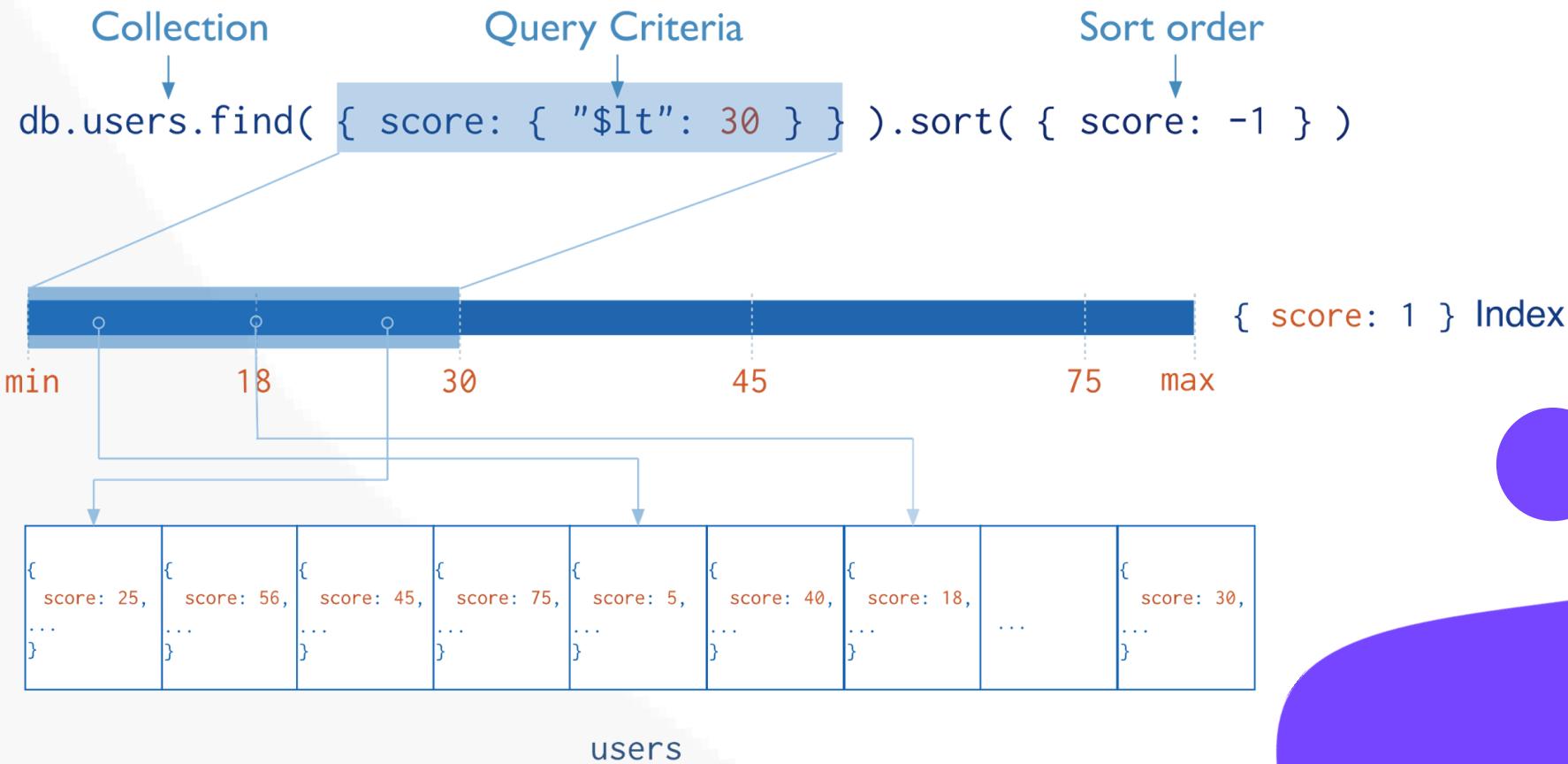
# Índices

IGTI



# Índices

IGTI



# Default Index

- Por default, o MongoDB cria um **índice único** no campo `_id`, durante a criação de uma collection;
- O **índice `_id`** evita que os clientes insiram dois documentos com o mesmo valor para o campo `_id`;
- Não é possível apagar esse índice.

```
> db.Equipamentos.insert({ "nome_equipamento": 'MAC0001', "data_insercao": new Date() });
WriteResult({ "nInserted" : 1 })
> db.Equipamentos.getIndexes();
[ { "v" : 2, "key" : { "_id" : 1 }, "name" : "_id_" } ]
>
>
```

# Single Field Index

- Índice com apenas um campo;
- Pode ser ascendente ou descendente;
- Criado com o comando: ***db.nome\_collection.createIndex ( { campo : N } )***
  - Onde **N** é a ordem de classificação do índice:
    - 1 → ascendente
    - -1 → descendente

```
db.records.createIndex( { score: 1 } )
```

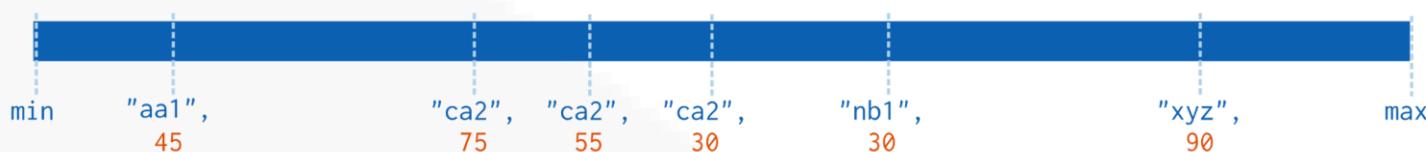


{ score: 1 } Index

# Compound Index

- Índice com vários campos;
- Cada campo pode ter sua ordem específica (ascendente ou descendente);
- Criado com o comando: ***db.nome\_collection.createIndex ( { campo1 : N, campo2 : N... } )***
  - Onde **N** é a ordem de classificação do índice:
    - 1 → ascendente
    - -1 → descendente

***db.records.createIndex( { userid: 1, score: -1 } )***



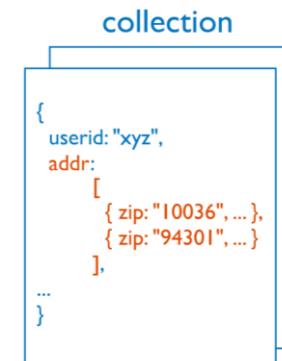
# Multikey Index

- Para indexar o conteúdo armazenado em campos do tipo **arrays**:
  - Faz a comparação com base em um ou mais elementos do array.
- O MongoDB cria entradas de índice separadas para cada elemento do array;
- Criado com o comando: ***db.nome\_collection.createIndex ( { campoarray : N } )***
  - Onde **N** é a ordem de classificação do índice:
    - **1 → ascendente**
    - **-1 → descendente**

***db.records.createIndex( { addr: 1 } )***



**{ "addr.zip": 1 } Index**



# Text Index

- Suporta a pesquisa de conteúdo de string em uma collection;
- Esses índices de texto não armazenam “stop words” específicas de um idioma (por exemplo, “o”, “a”, “ou”);
  - Armazena apenas palavras raiz → “**Curso de Bootcamp do IGTI**”.
- Criado com o comando: ***db.nome\_collection.createIndex ( { campoarray : “text” } );***
- Pode ser simples ou composto.

```
db.reviews.createIndex(  
  {  
    subject: "text",  
    comments: "text"  
  }  
)
```

```
db.reviews.createIndex( { comments: "text" } )
```

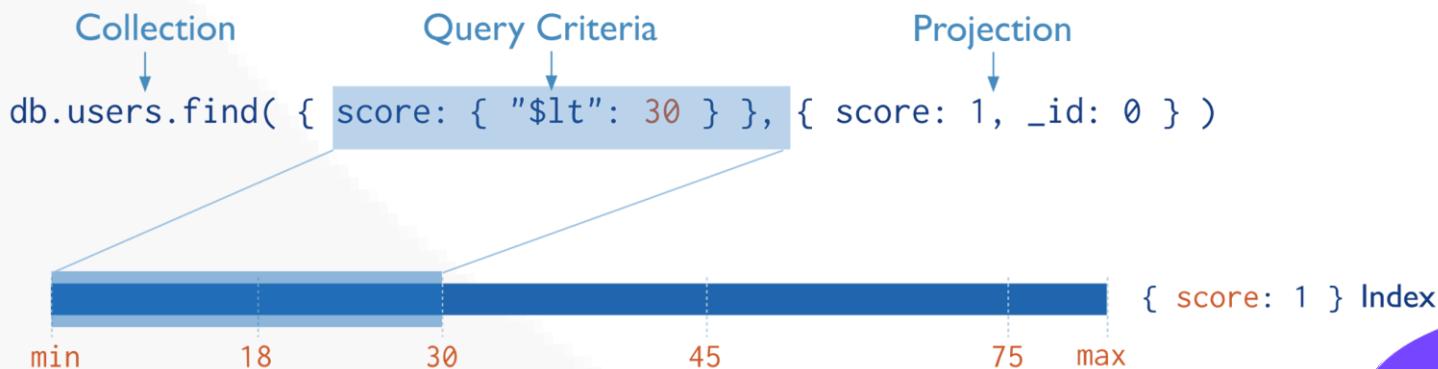
# Partial Index

- Os índices parciais indexam apenas os documentos em uma collection que **atendem a uma expressão de filtro especificada**:
  - Mais performático → busca em conjunto menor de dados (estrutura menor do índice)
  - Gasta menos espaço de armazenamento;
  - Menor custo na criação e manutenção.

```
db.restaurants.createIndex(  
  { cuisine: 1, name: 1 },  
  { partialFilterExpression: { rating: { $gt: 5 } } }  
)
```

# Covered Query

- Consulta **totalmente coberta pelo índice**;
- Quando os critérios de filtro (“where”) e a projeção (“select”) de uma query incluem apenas o(s) campo(s) indexado(s);
  - O MongoDB retorna os resultados diretamente do índice → muito eficiente.



# Index Intersection

- O MongoDB pode usar a interseção de vários índices para atender às queries;
- Em geral, cada intersecção envolve dois índices;
  - O MongoDB pode empregar várias intersecções aninhadas de índice para resolver uma query.

```
{ qty : 1 }  
{ status : 1 , ord_date : -1 }
```

```
db . pedidos . find ( { qty : { $ gt : 10 } , status : "A" } )
```

# Próxima Aula



- ❑ Plano de Execução de Query.

# Otimização de Query no MongoDB

---

CAPÍTULO 7. AULA 7.2. PLANO DE EXECUÇÃO DE QUERY

PROF. GUSTAVO AGUILAR

# Nesta Aula



- ❑ Plano de Execução de Query no MongoDB.
- ❑ Método *explain()*.

# Plano de Execução de Query



# Demo



A presentation slide about MongoDB is displayed on a computer monitor. The slide has a teal header and footer and a white central area. It features several sections with icons and text:

- SGBD e SGBDD**  
Banco de dados centralizado / distribuído  
Multiplataforma  
Escrito em C++  

- Lançado em 02/2009**  
Inicialmente 10gen.  
Atualmente ©MongoDB Inc.  
 
- Gratuito e Open Source**  
Edição Community Server  
\*Edição Enterprise  
(subscrição com suporte)  

- NOSQL** “*Not Only SQL*”  
Não relacional  

- Orientado a Documento**  
Esquema Dinâmico  
*Document Database*  
Formato JSON (field-value)  
Armazenado como BSON  

- Universidade MongoDB**  
Cursos gratuitos e online, em inglês, com certificado  
<https://university.mongodb.com>  


The monitor sits on a white desk against a teal background. The IGTI logo is visible in the top right corner.

# Próxima Aula



- ❑ Otimização de Operações CRUD.

# Otimização de Query no MongoDB

---

CAPÍTULO 7. AULA 7.3. OTIMIZAÇÃO DE OPERAÇÕES CRUD

PROF. GUSTAVO AGUILAR

# Nesta Aula



- ❑ Otimização de Operações CRUD na prática.

# Otimização de Operações CR



# Demo



A computer monitor displays an advertisement for MongoDB. The ad features a large green leaf icon and the text "mongoDB®". The content is organized into several sections:

- SGBD e SGBDD**  
Banco de dados centralizado / distribuído  
Multiplataforma  
Escrito em C++  

- Lançado em 02/2009**  
Inicialmente 10gen.  
Atualmente ©MongoDB Inc.  

- Gratuito e Open Source**  
Edição Community Server  
\*Edição Enterprise  
(subscrição com suporte)  

- NOSQL**  
“Not Only SQL”  
Não relacional  

- Orientado a Documento**  
Esquema Dinâmico  
*Document Database*  
Formato JSON (field-value)  
Armazenado como BSON  

- Universidade MongoDB**  
Cursos gratuitos e online, em inglês, com certificado  
<https://university.mongodb.com>  


# Otimização de Operações CRUD

```
db.restaurants.find({ "address.zipcode": { $gt: '50000' }, cuisine: 'Sushi' })  
    .sort({ stars: -1 })
```

Index	None	{ "address.zipcode": 1, "cuisine": 1, "stars": 1 }	{ "cuisine": 1, "address.zipcode": 1, "stars": 1 }	{ "cuisine": 1, "stars": 1, "address.zipcode": 1 }
Exec Time	386 ms	279 ms	90 ms	43 ms
Docs Returned	11,611	11,611	11,611	11,611
Docs Examined	1,000,000	11,611	11,611	11,611
Keys Examined	0	95,988	11,611	11,663
In-Memory Sort	yes	yes	yes	no

# Próxima Aula



- ❑ Capítulo 8 - Operações Massivas de Dados.

# Performance e Otimização

Capítulo 8. Operações Massivas de Dados

PROF. GUSTAVO AGUILAR

# Operações Massivas de Dados

---

CAPÍTULO 8. AULA 8.1. EXTRAÇÃO E CARGA MASSIVA DE DADOS NO SQL SERVER

PROF. GUSTAVO AGUILAR

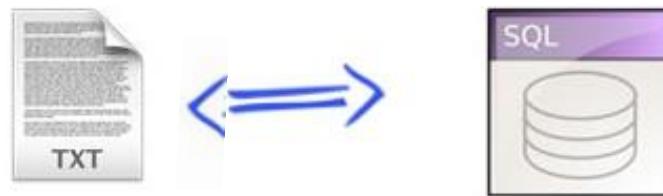
# Nesta Aula



- Utilitário BCP.
- Bulk Insert.
- Integration Services (SSIS).
- Azure Data Factory.

# Utilitário BCP

- Ferramenta de linha de comando (**bcp.exe**), para exportar e importar dados entre um banco de dados SQL Server e um arquivo de dados:
  - ✓ Exportar de uma tabela do SQL Server para um arquivo de dados;
  - ✓ Exportar para um arquivo de dados com base em uma consulta;
  - ✓ Importar dados de um arquivo de dados para uma tabela do SQL Server.



# Utilitário BCP

- Exportar dados de uma tabela para um arquivo, usando autenticação integrada:

```
bcp AdventureWorks2012.Sales.Currency out "Currency Types.dat" -T -c
```

- Exportar dados de uma query para um arquivo:

```
bcp "SELECT FullName, PreferredName FROM WideWorldImporters.dbo.People  
ORDER BY FullName" queryout D:\BCP\People.txt -t, -c -T
```

- Importar dados de um arquivo para uma tabela:

```
bcp BDTeste.dbo.Tabela_A_Ser_Carregada IN
```

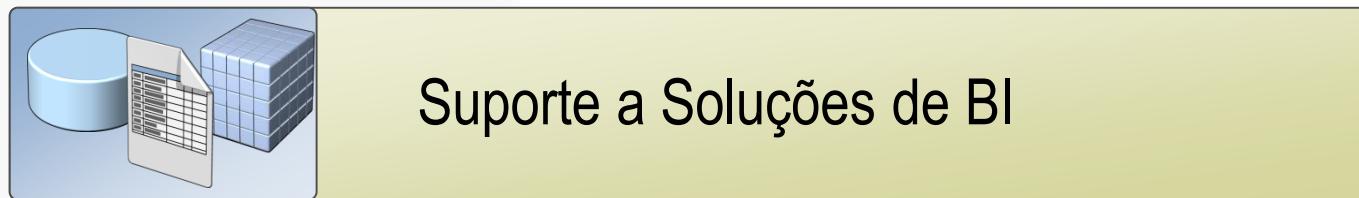
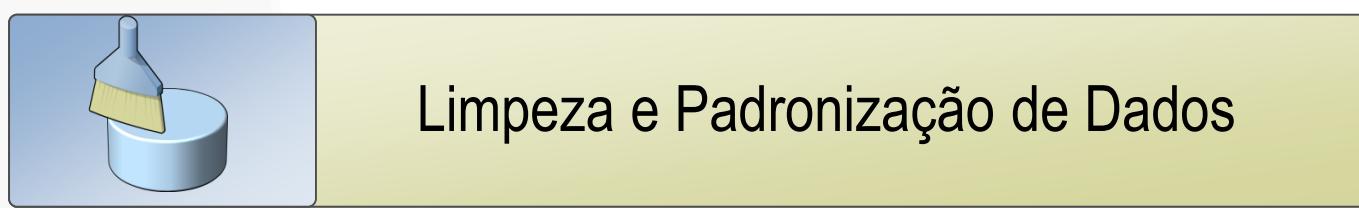
```
D:\BCP\StockItemTransactions_character.bcp -S Nome_Servidor -c -T
```

# Bulk Insert

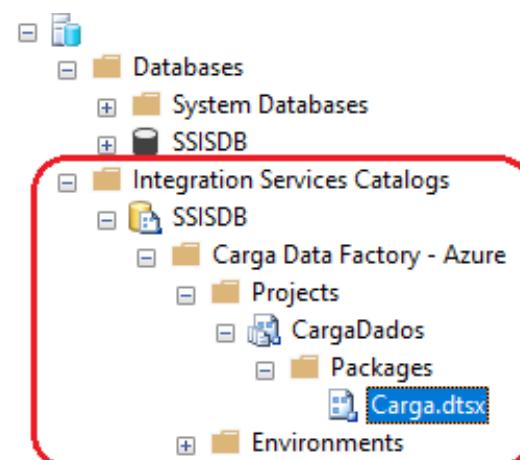
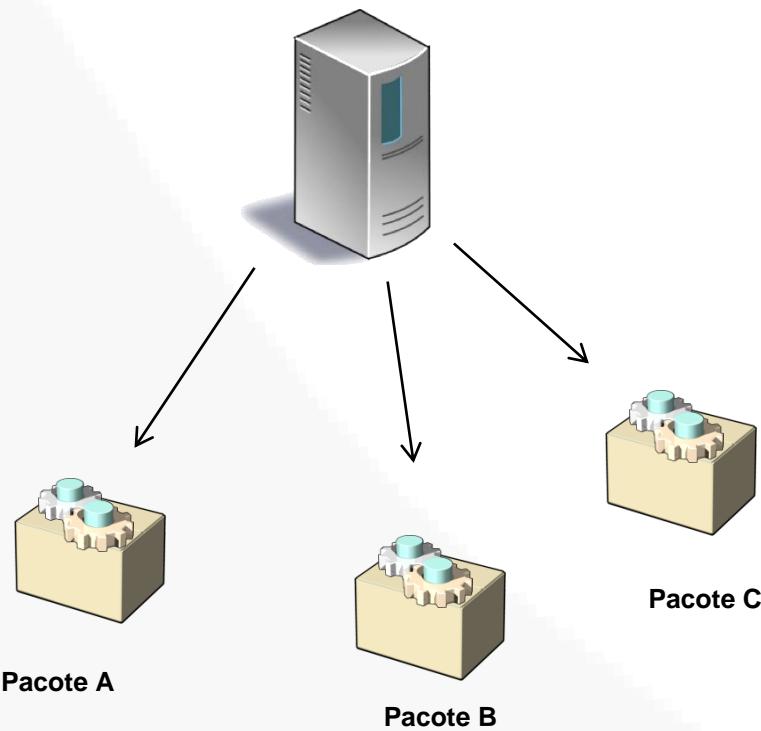
- Operação de carga de dados, possível de ser minimamente logada\*:
  - ✓ Menos espaço gasto e contenção no transaction log;
  - ✓ Mais otimizada por não registrar cada insert como uma transação\*.

```
BULK INSERT BDProducoes.dbo.NOME_DA_TABELA  
FROM 'C:\PATHNOME_DO_ARQUIVO.txt'  
WITH ( FIELDTERMINATOR ='\t',  
      ROWTERMINATOR ='\n',  
      );
```

# Integration Services (SSIS)

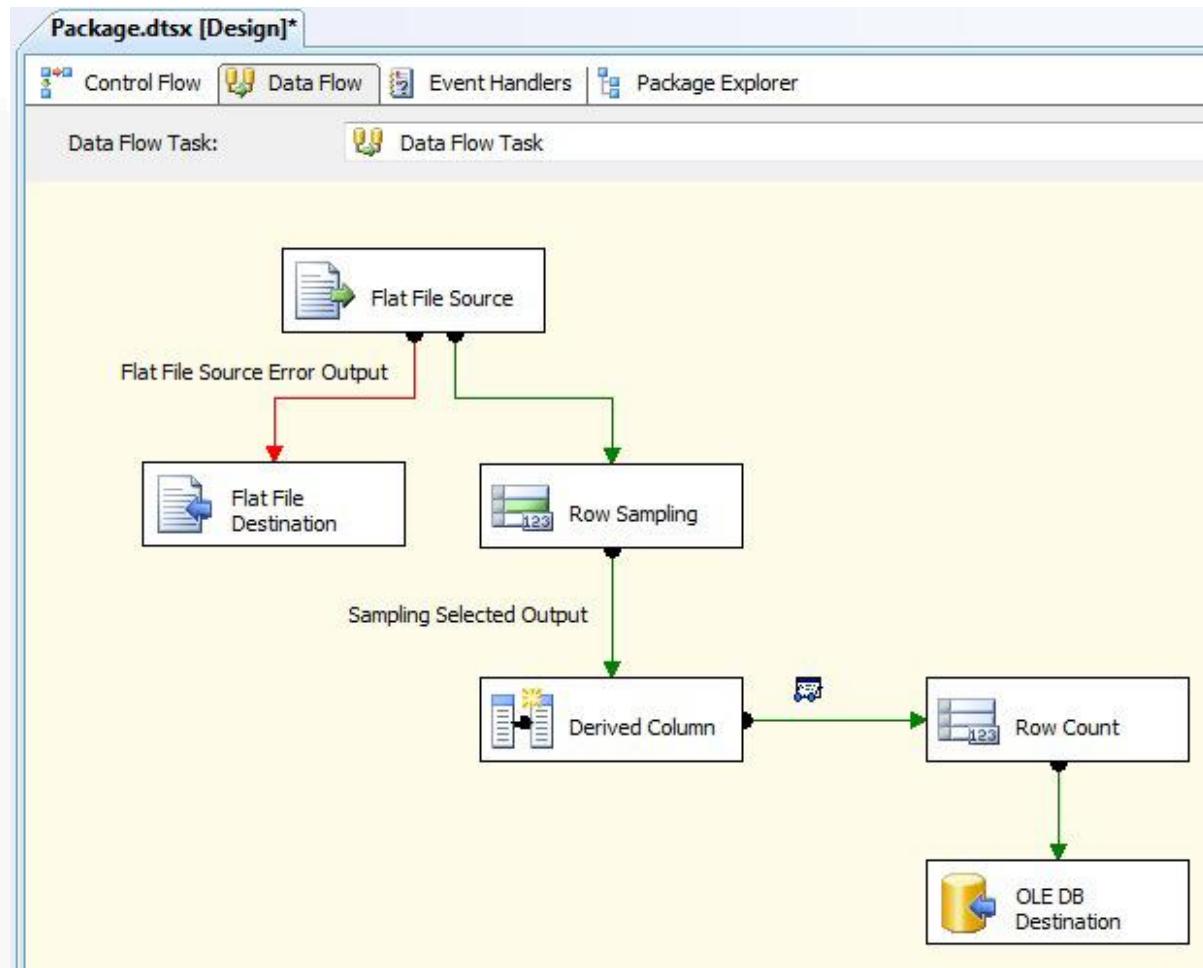


# Integration Services (SSIS)



# Integration Services (SSIS)

IGTi



# Azure Data Factory

- Serviço de integração de dados e ETL baseado em nuvem;
- Conectores para mais de 90 tipos de fontes de dados diferentes;
- ETLs simples à complexos, com integração com Azure HDInsight, Azure Databricks ou Azure Synapse Analytics.



FTP server



Data Factory

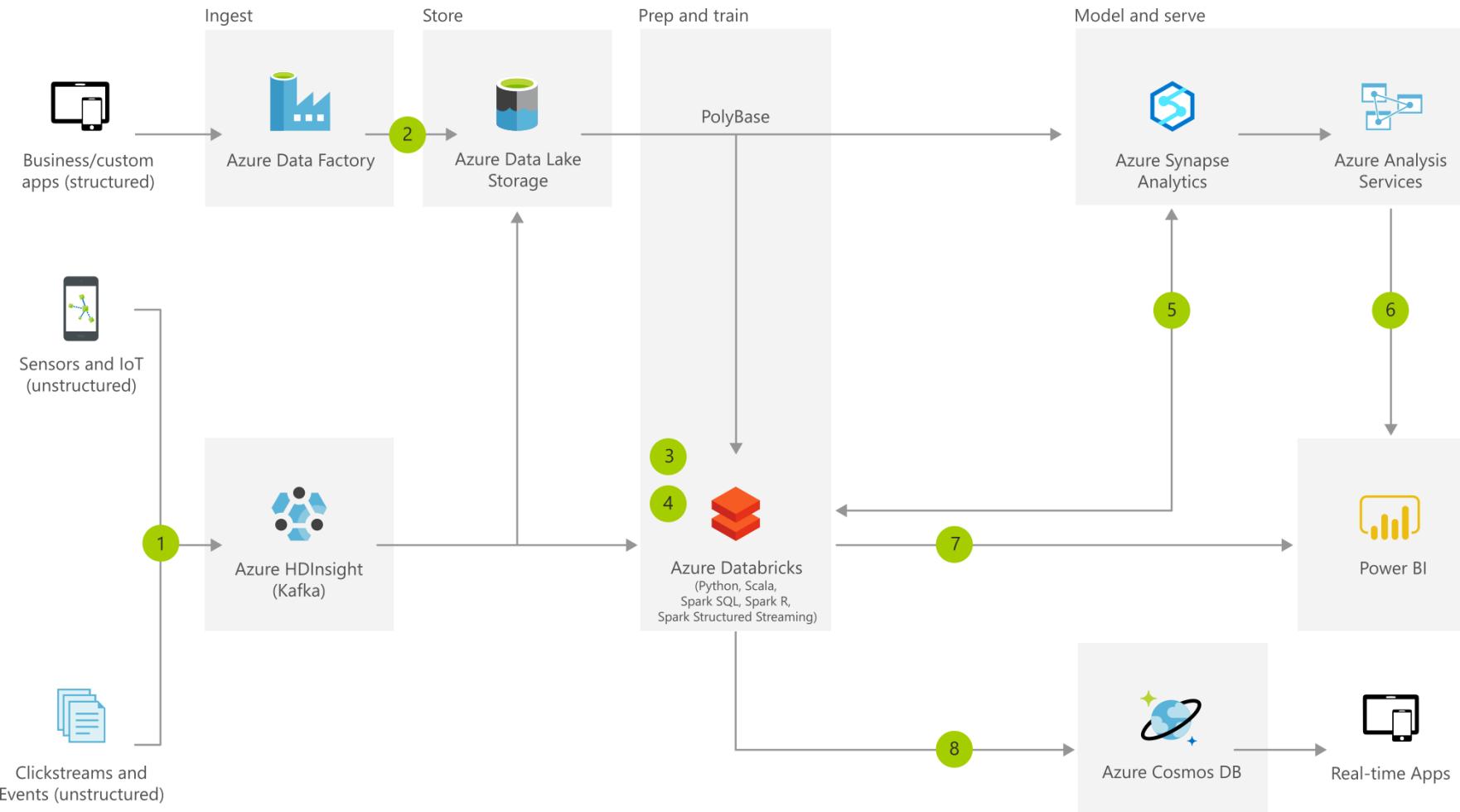


Blob Storage



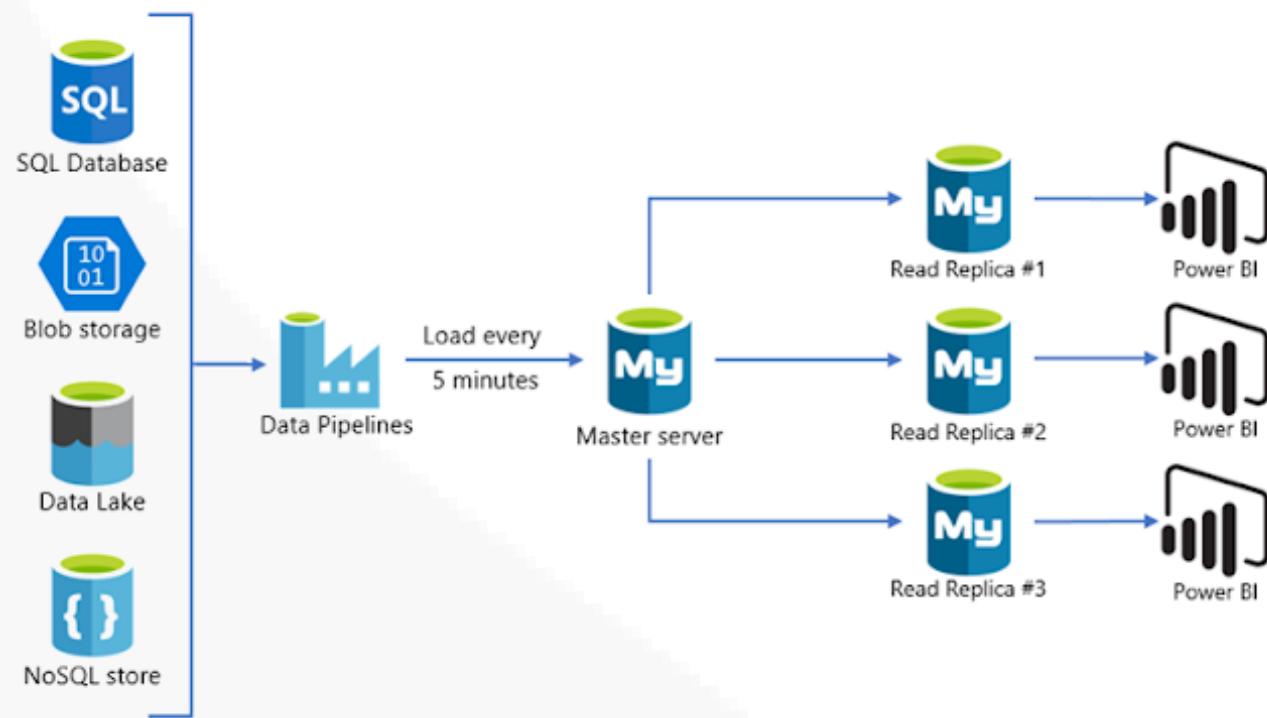
# Azure Data Factory

IGTI



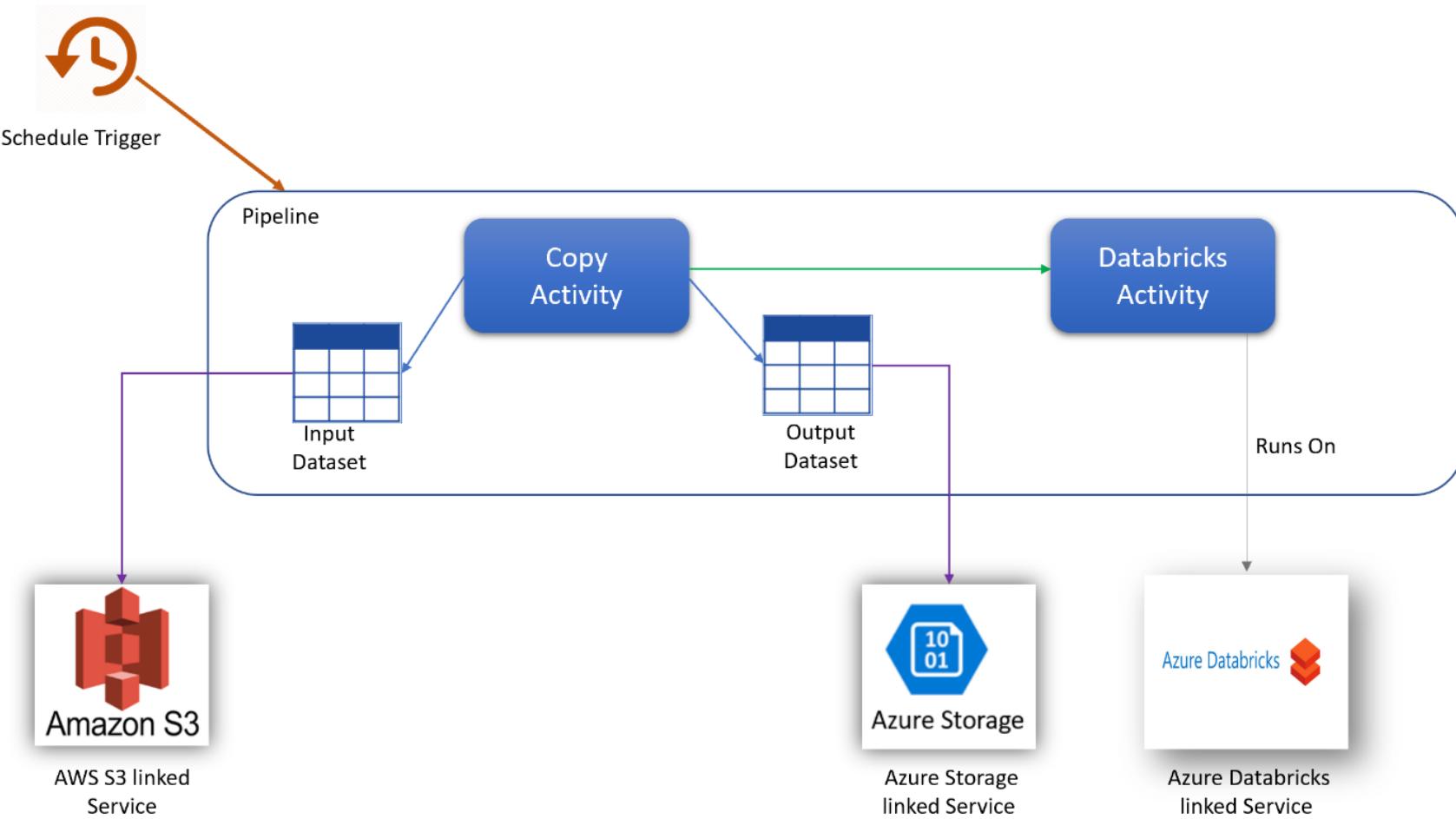
# Azure Data Factory

- Permite criar e agendar fluxos de trabalho orientados a dados para orquestrar a movimentação de dados e transformá-los;



# Azure Data Factory

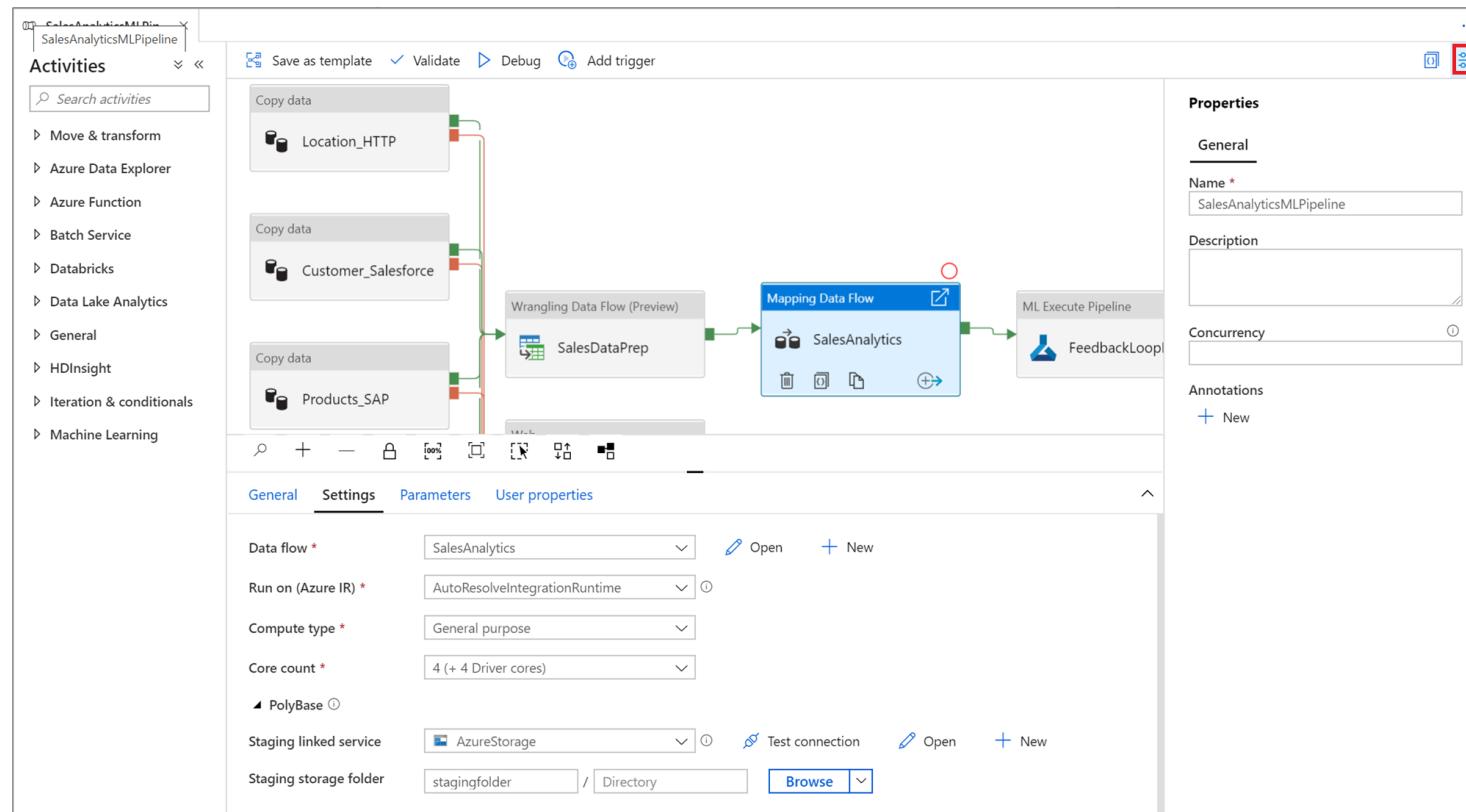
IGTI



# Azure Data Factory

IGTI

- Possibilita a criação de pipelines de forma gráfica ou via código.



# Azure Data Factory

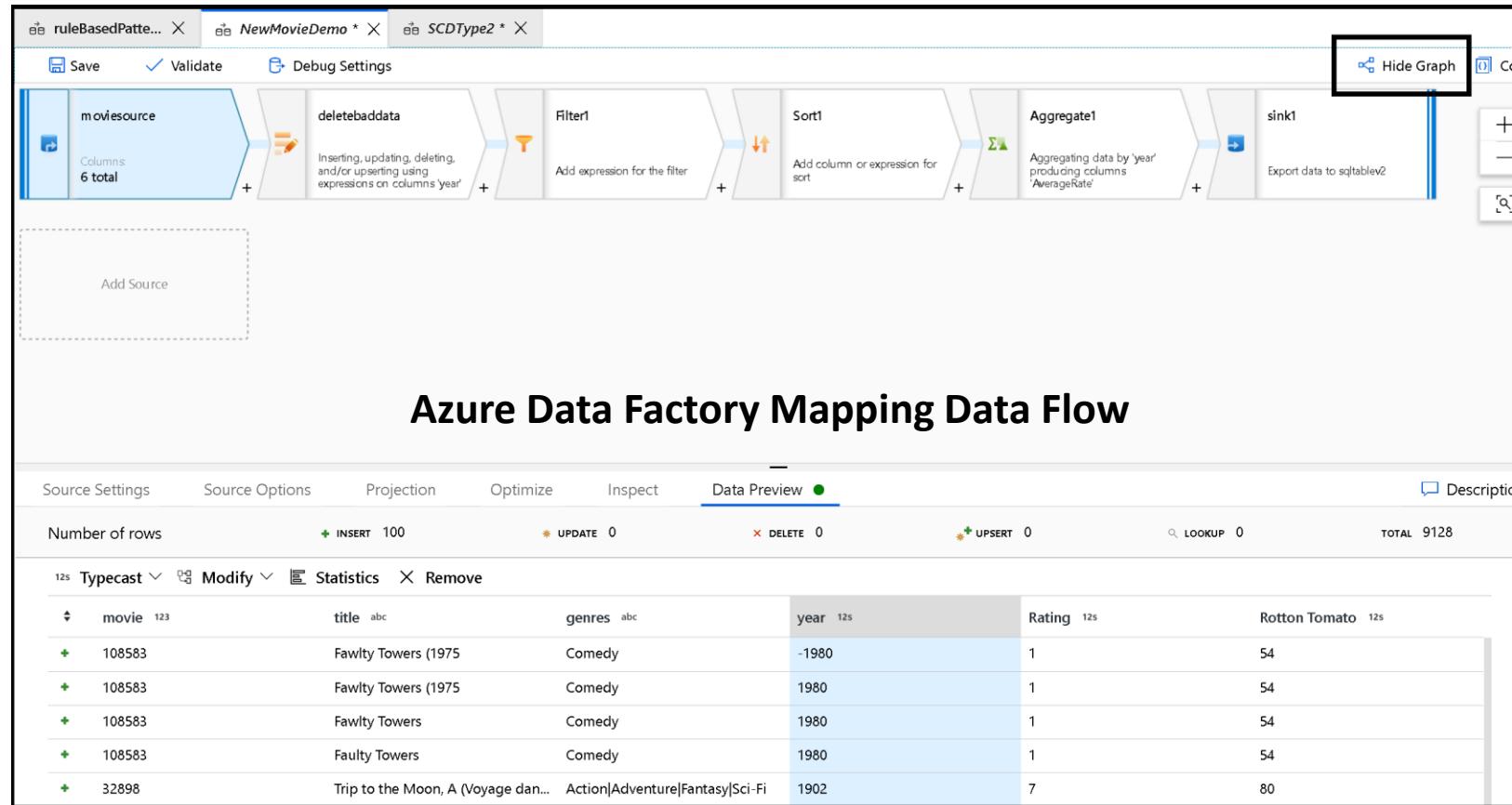


## MAPEAMENTO DE FLUXO DE DADOS (MAPPING DATA FLOW)

- São transformações de dados visualmente projetadas no Azure Data Factory;
- Permitem que os engenheiros de dados desenvolvam lógicas de transformação de dados sem escrever código;
- O Data Factory executará a lógica em um cluster Spark, autogerenciado pelo Azure, que será ativado e desativado quando necessário.

# Azure Data Factory

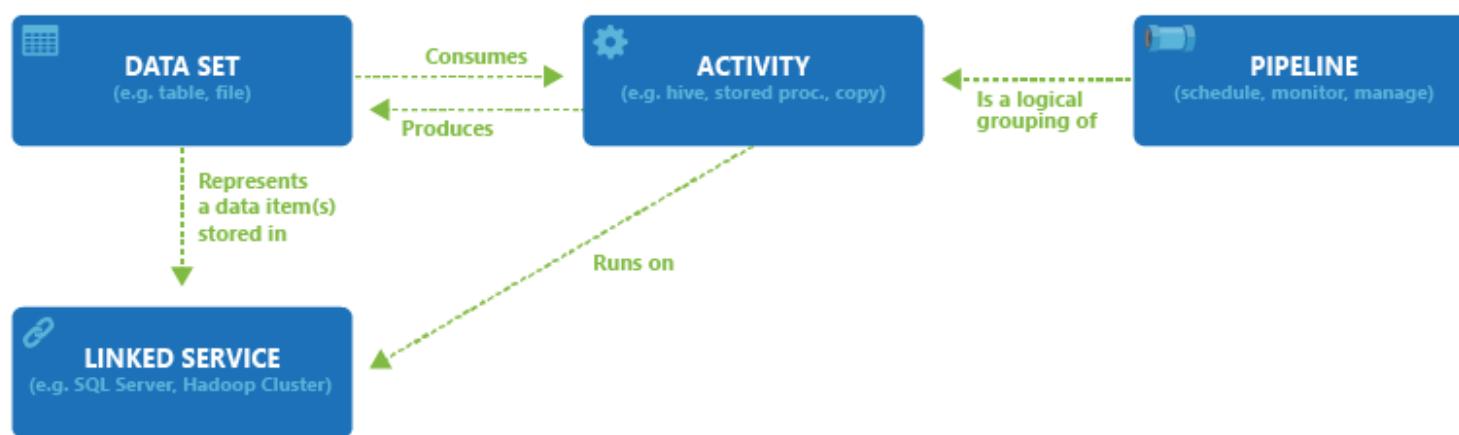
IGTI



# Azure Data Factory

IGTI

- Linked Service
- Data set
- Atividades
- Pipeline



# Próxima Aula



- ❑ Extração e Carga Massiva de Dados no MongoDB.

# Operações Massivas de Dados

---

CAPÍTULO 8. AULA 8.2. EXTRAÇÃO E CARGA MASSIVA DE DADOS NO MONGODB

PROF. GUSTAVO AGUILAR

# Nesta Aula



- ❑ Operações de BulkWrite
- ❑ mongoimport.
- ❑ mongoexport.
- ❑ mongorestore.
- ❑ mongodump.

# Operações de Bulk Write

```
db.collection.bulkWrite(  
  [  
    { insertOne : <document> },  
    { updateOne : <document> },  
    { updateMany : <document> },  
    { replaceOne : <document> },  
    { deleteOne : <document> },  
    { deleteMany : <document> }  
  ]  
)  
  
try {  
  db.characters.bulkWrite([  
    { insertOne: { "document": { "_id": 4, "char": "Dithras", "class": "barbarian", "lvl": 4 } } },  
    { insertOne: { "document": { "_id": 4, "char": "Taeln", "class": "fighter", "lvl": 3 } } },  
    { updateOne : {  
      "filter" : { "char" : "Eldon" },  
      "update" : { $set : { "status" : "Critical Injury" } }  
    },  
    { replaceOne : {  
      "filter" : { "char" : "Meldane" },  
      "replacement" : { "char" : "Tanys", "class" : "oracle", "lvl": 4 }  
    }  
  ], { ordered : false } );  
} catch (e) {  
  print(e);  
}
```

# mongoimport

- Ferramenta de linha de comando para importar dados de arquivos em formatos JSON , CSV ou TSV para collections em um banco de dados MongoDB.

***mongoimport --db = users --collection = contacts --file = contacts.json***

# mongoexport

- Ferramenta de linha de comando para exportar dados de collections para arquivos nos formatos JSON ou CSV.

***mongoexport --collection = events --db = reporting --out = events.json***

# mongodump

- Ferramenta de linha de comando para gerar uma exportação binária do conteúdo de um banco de dados MongoDB.

***mongodump --archive = test.20150715.gz --gzip --db = test***

# mongorestore

- Ferramenta de linha de comando para carregar dados de um dump binário de banco de dados MongoDB criado com o mongodump.

```
mongorestore --db=reporting dump/test/salaries.bson
```

# Próxima Aula



- ❑ Expurgo Massivo de Dados no SQL Server.

# Operações Massivas de Dados

---

CAPÍTULO 8. AULA 8.3. EXPURGO MASSIVO DE DADOS NO SQL SERVER

PROF. GUSTAVO AGUILAR

# Nesta Aula



- Switch Table.
- Tabela Particionada.
- Estratégia Shift-Table.

# Switch Table

- Mover dados de uma tabela para outra de forma rápida;
- Remove dados da origem, arquivando os dados no destino.



Before switch:

2012-01-01	...	...
2012-12-31	...	...
2013-01-01	...	...
2013-12-31	...	...
2014-01-01	...	...
2014-12-31	...	...
2015-01-01	...	...
2015-12-31	...	...

A dashed red box highlights the data from the **Billing.price** table before the switch, which is then moved to a separate table. An arrow points from the last row of the original table to this new table, indicating the archiving process.

After switch:

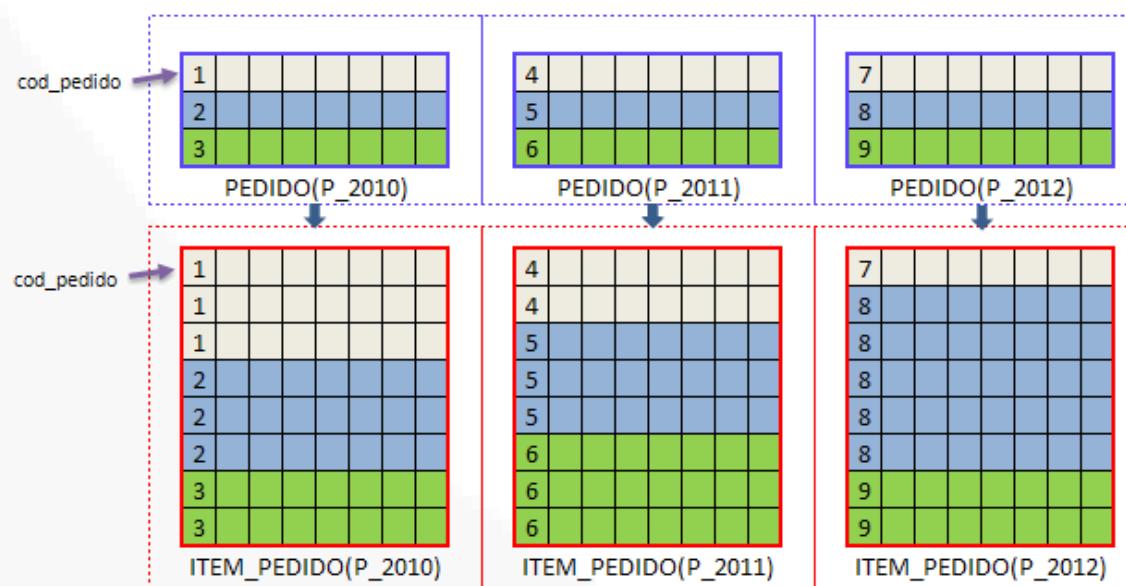
The **Billing.price** table is now empty, indicated by a dashed red border around the table structure.

The archived data from the **Billing.price** table is now stored in the **Billing.rate** table, shown as a red-bordered section of the table.

*ALTER TABLE TabelaOrigem SWITCH TO TabelaDestino;*

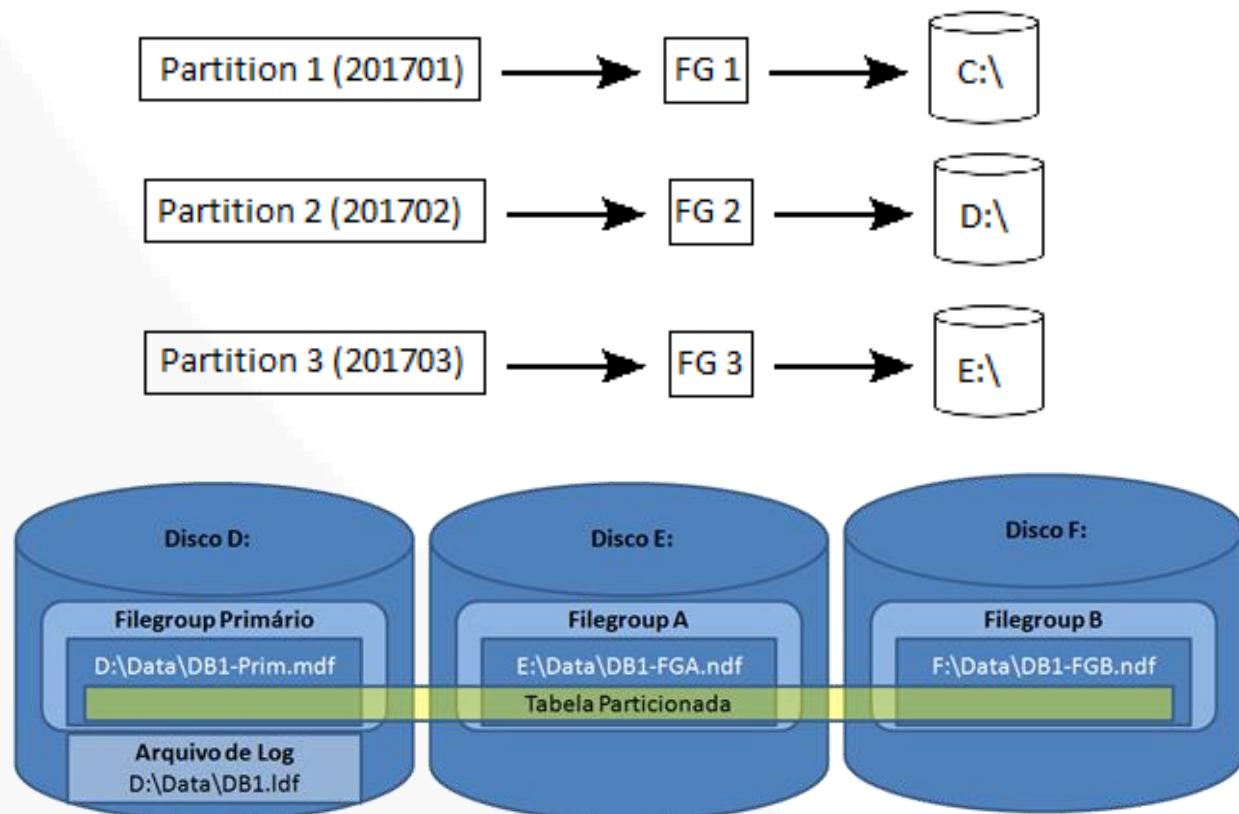
# Tabela Particionada

- Recurso que permite a distribuição dos dados de tabelas e índices em vários grupos, separados fisicamente.



# Tabela Particionada

- Essa distribuição pode ser feita para discos físicos distintos.



# Tabela Particionada

- **Função de particionamento:** define como as linhas são distribuídas para um conjunto de partições, com base nos valores de certas colunas, chamadas de colunas de particionamento.

2012-01-01	...	...
2012-12-31	...	...
2013-01-01	...	...
2013-12-31	...	...
2014-01-01	...	...
2014-12-31	...	...
2015-01-01	...	...
2015-12-31	...	...

# Tabela Particionada

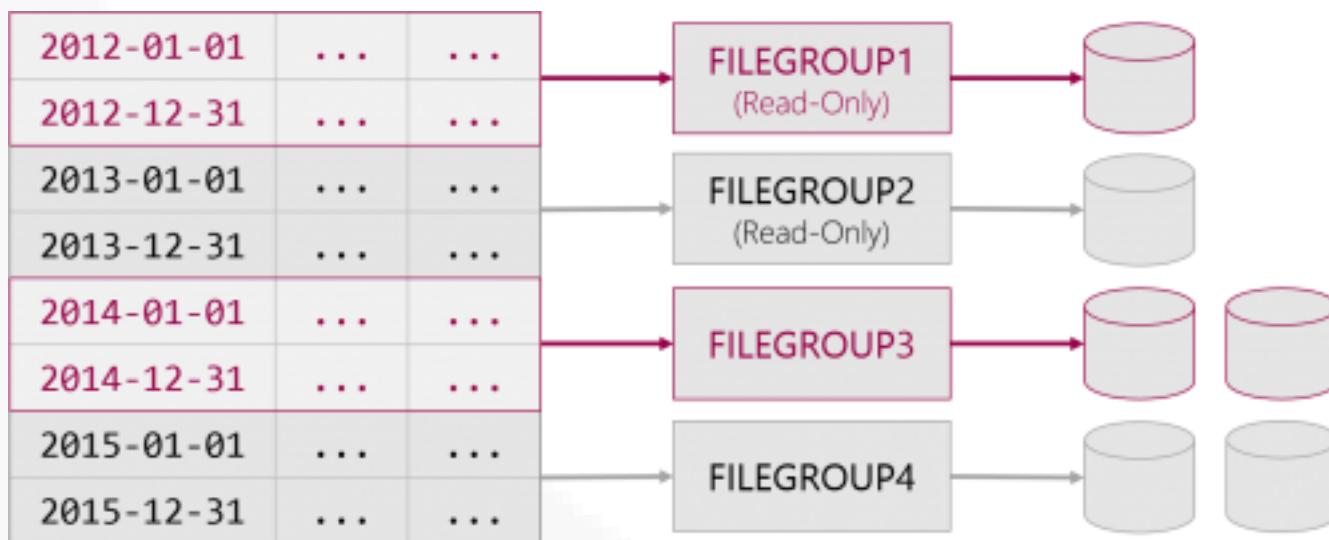
- Ex: criar uma função de particionamento de nome FP\_Particiona\_Tabela que irá particionar a tabela em 4 partições:

```
CREATE PARTITION FUNCTION FP_Particiona_Tabela (int)
AS RANGE LEFT FOR VALUES (1, 100, 1000) ;
```

Partition	1	2	3	4
Valores	<code>col1 &lt;= 1</code>	<code>col1 &gt; 1 AND col1 &lt;= 100</code>	<code>col1 &gt; 100 AND col1 &lt;= 1000</code>	<code>col1 &gt; 1000</code>

# Tabela Particionada

- **Esquema de particionamento** mapeia cada partição especificada pela função de particionamento para um filegroup.



# Tabela Particionada

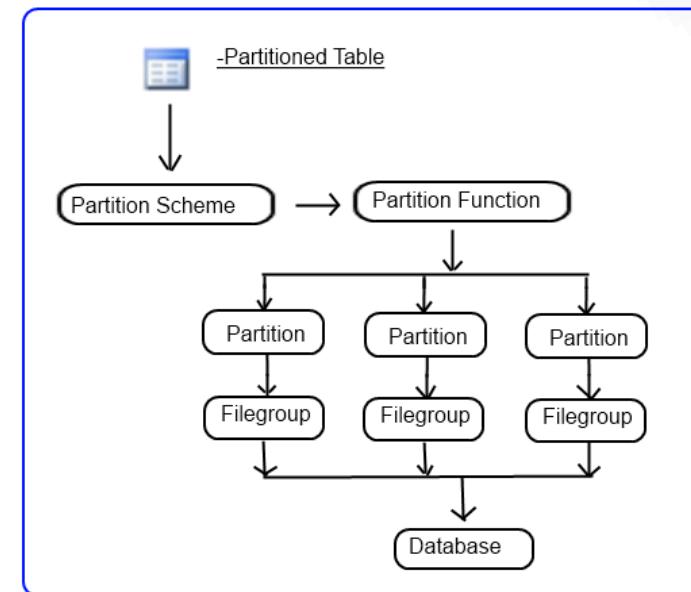
- Ex.: criar o esquema de partição chamado PS\_Particoes que direciona as partições para os filegroups específicos:
  - De-para das faixas de valores da função de partição com os filegroups

```
CREATE PARTITION SCHEME PS_Particoes  
AS PARTITION FP_Particiona_Tabela  
TO (Filegroup1, Filegroup2, Filegroup3, Filegroup4);
```

# Tabela Particionada

- Criar a tabela particionada apontando para o esquema de particionamento.

```
CREATE TABLE TabelaParticionada  
(col1 int PRIMARY KEY, col2 char(10))  
ON PS_Particoes (col1);
```



# Tabela Particionada

- Arquivar uma partição para uma tabela não particionada;
- Expurgar uma partição.

A diagram illustrating the movement of partitions. On the left, a table structure shows two rows: '2012-01-01' and '2012-12-31'. An arrow points from this table to a dashed-line box representing a non-partitioned table structure. The table has three columns and four rows, with the first two rows ('2012-01-01' and '2012-12-31') corresponding to the ones in the original table.

2012-01-01	...	...
2012-12-31	...	...
2013-01-01	...	...
2013-12-31	...	...
2014-01-01	...	...
2014-12-31	...	...
2015-01-01	...	...
2015-12-31	...	...

After switch:

A diagram illustrating the result of the partition switch operation. A dashed-line box represents the state after the switch. It contains the row '2012-01-01' and '2012-12-31' from the original table, which are now in a separate table structure. The main table structure below it contains the rows from 2013 onwards: '2013-01-01', '2013-12-31', '2014-01-01', '2014-12-31', '2015-01-01', and '2015-12-31'.

2013-01-01	...	...
2013-12-31	...	...
2014-01-01	...	...
2014-12-31	...	...
2015-01-01	...	...
2015-12-31	...	...

2012-01-01	...	...
2012-12-31	...	...

```
TRUNCATE TABLE Sales.TransactionHistory  
WITH (PARTITIONS (12, 14, 16 TO 18))  
GO
```

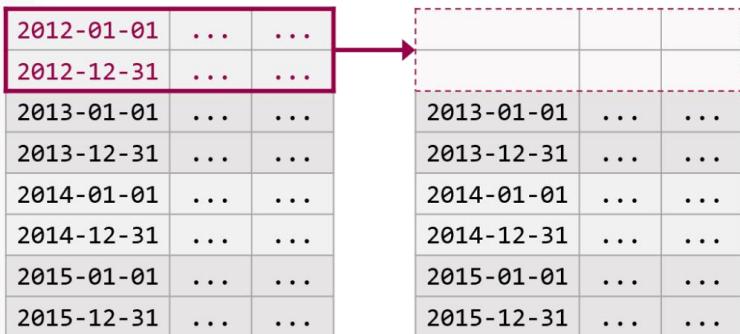
SQL 2016+

*ALTER TABLE TabelaOrigem SWITCH  
PARTITION 1 TO TabelaTarget;*

# Tabela Particionada

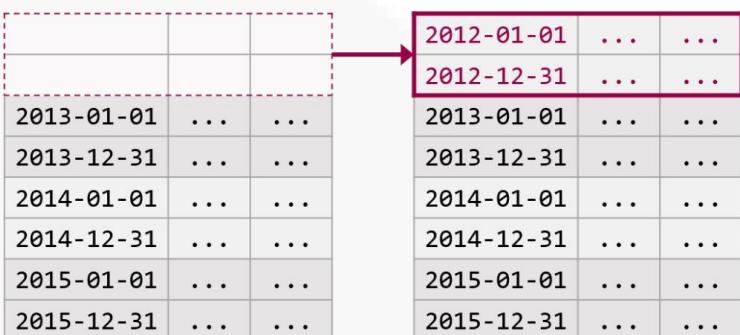
- Arquivar uma partição para uma partição de outra tabela particionada.

Before switch:



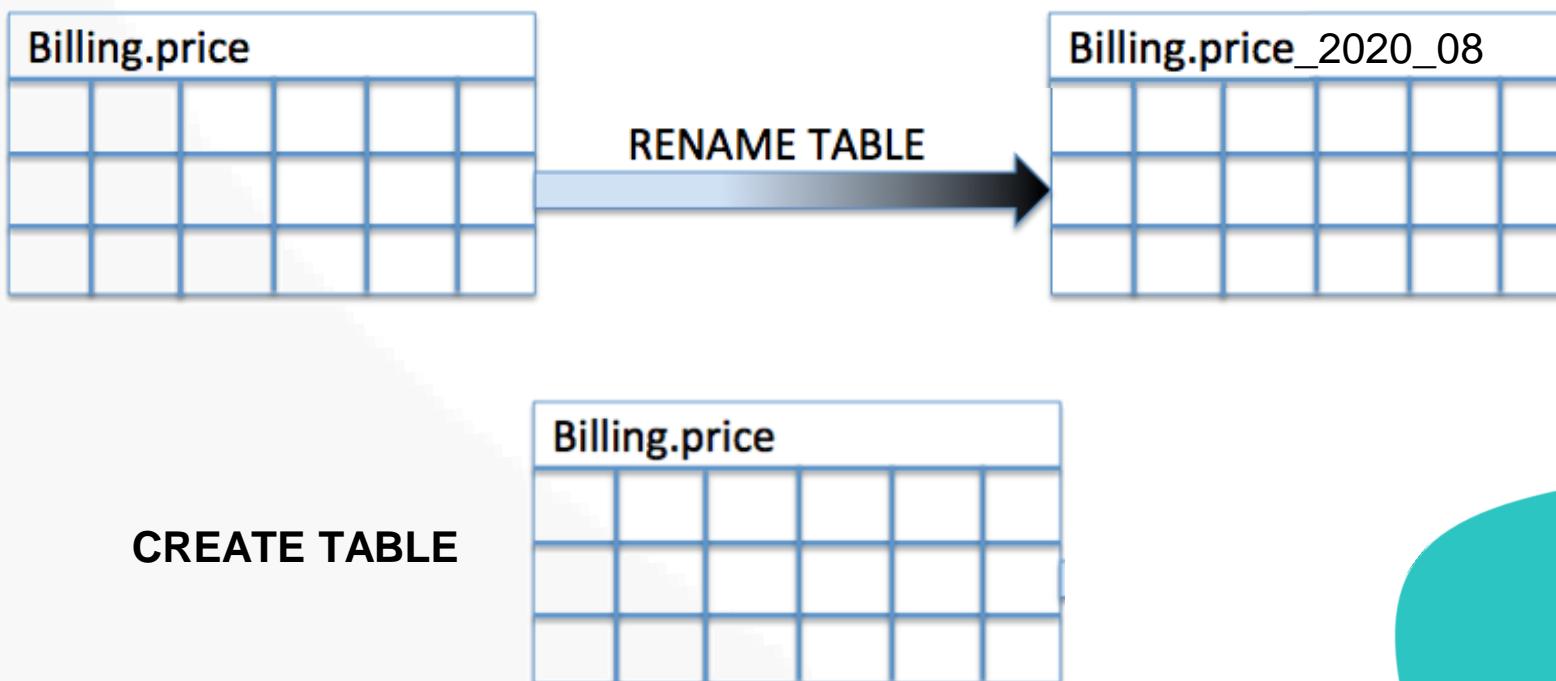
*ALTER TABLE Source  
SWITCH PARTITION 1  
TO Target PARTITION 1*

After switch:



# Estratégia Shift Table

- Arquivar / Expurgar;
- Dados a expurgar > dados remanescentes.



# Próxima Aula



- ❑ Expurgo Massivo de Dados no MongoDB.

# Operações Massivas de Dados

---

CAPÍTULO 8. AULA 8.4. EXPURGO MASSIVO DE DADOS NO MONGODB

PROF. GUSTAVO AGUILAR

# Nesta Aula



- Operações de BulkWrite
- Índice TTL.
- Shift-Collection.

# Operações de Bulk Write

```
db.collection.bulkWrite(  
  [  
    { insertOne : <document> },  
    { updateOne : <document> },  
    { updateMany : <document> },  
    { replaceOne : <document> },  
    { deleteOne : <document> },  
    { deleteMany : <document> }  
  ]  
)  
  
try {  
  db.characters.bulkWrite([  
    { insertOne: { "document": { "_id": 4, "char": "Dithras", "class": "barbarian", "lvl": 4 } } },  
    { insertOne: { "document": { "_id": 5, "char": "Taeln", "class": "fighter", "lvl": 3 } } },  
    { updateOne : {  
      "filter" : { "char" : "Eldon" },  
      "update" : { $set : { "status" : "Critical Injury" } }  
    } },  
    { deleteOne : { "filter" : { "char" : "Brisbane" } } },  
    { deleteMany : { "filter" : { "encounter": { $lt : 0 } } } },  
    { replaceOne : {  
      "filter" : { "char" : "Meldane" },  
      "replacement" : { "char" : "Tanys", "class" : "oracle", "lvl": 4 }  
    } }  
  ]);  
} catch (e) {  
  print(e);  
}
```

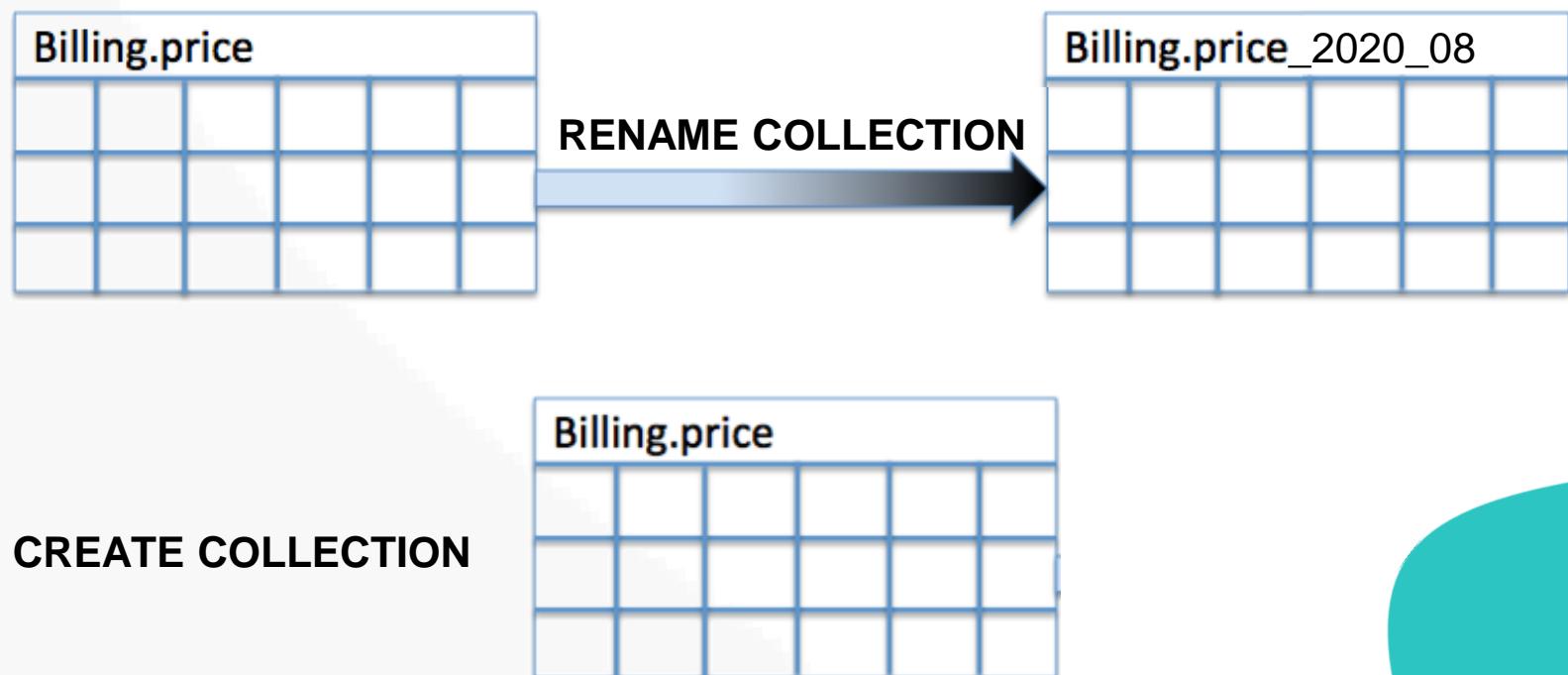
# Índices TTL

- Dados são “deletados” (expirados) logicamente;
- Deletados fisicamente pelo MongoDB, em background, sem sobrecarregar o ambiente.

```
db.eventlog.createIndex
(
    { "lastModifiedDate": 1 },
    { expireAfterSeconds: 3600 }
)
```

# Estratégia Shift Collection

- Arquivar / Expurgar;
- Dados a expurgar > dados remanescentes.





# Performance e Otimização

FIM

PROF. GUSTAVO AGUILAR