

# BANCOS DE DADOS: DEFINIÇÕES



Unidade 01



# Sumário

03	Introdução
04	Objetivos de Aprendizagem
05	1.1 Bancos de dados
06	1.1.1 SGBD
06	1.2 Bancos de dados Relacionais
07	1.2.1 Características de um Banco de dados Relacional
10	1.2.2 Quando usar um banco de dados relacional
11	1.3 Arquitetura de um SGBD Relacional
13	1.3.1 Utilitários do SGBD
13	1.3.2 Opções de implementação de um banco de dados relacional
15	1.4 Bancos de dados distribuídos
16	1.4.2 Questões para o controle de concorrência em ambientes distribuídos
16	1.4.3 Teorema CAP
18	1.5 Cloud – DbaaS: Banco de dados como um serviço
19	1.6 Bancos de dados NoSQL (Not Only Sql)
21	1.6.1 Tipos de bancos de dados NoSql
21	1.6.1.1 Orientados a documentos
22	1.6.1.2 Chave-valor
23	1.6.1.4 Baseados em grafos
23	1.6.2 Vantagens de um banco de dados NoSql
24	1.7 Bancos de dados promissores
24	1.7.1 Amazon Web Services: DynamoDB
25	1.7.2 Google: BigQuery e BigTable
26	1.7.3 Microsoft: Cosmos DB
27	1.7.4 Cockroach DB
28	1.8 Outros tipos de banco de dados
30	Síntese
31	Fullture Insight

# Olá, **Fullturist,**

**Seja bem-vindo(a)!**

## **BANCOS DE DADOS: DEFINIÇÕES**

---

Iremos acompanhá-lo(a) no desenvolvimento desta Trilha de Aprendizagem, a qual vai trazer informações a respeito da tecnologia e do ambiente de negócios em bancos de dados.

Pode ser surpreendente notar que este é um tema muito importante em nossa vida atualmente. Praticamente todos os eventos de nossa vida passada, e até mesmo de nossas tendências futuras, estão registrados em um banco de dados. Desde informações de nossa vida civil, como documentos pessoais e registros de propriedades, nossa vida pessoal, através de redes sociais, e até mesmo nosso trajeto ao trabalho são registrados em um banco de dados.

Como você pode perceber, assim como esse tema pode ser vasto em nossa vida cotidiana, a tecnologia oferecida para atender essas diversas situações também é ampla. Entretanto, embora as opções tecnológicas sejam numerosas, esta trilha fornecerá a você todos os conhecimentos para entender o ambiente tecnológico e de negócios disponíveis para a solução da maioria das necessidades. Além disso, conheceremos o que há por trás de toda essa tecnologia e como todas essas informações são armazenadas e recuperadas!

Também abordaremos temas como a organização e execução de um projeto de banco de dados, uma linguagem para acesso aos dados armazenados (PL/SQL da Oracle) e, ainda, importantes questões que impactam em muito o desempenho de uma aplicação.  
Bom estudo!

# Unidade 1

## Olá, Fullturist!

Esta é a Unidade 1 da Trilha de Aprendizagem Bancos de Dados. Nesta unidade você terá a oportunidade conhecer toda uma gama de opções em armazenamento, gerenciamento de dados e suas melhores aplicações. O foco principal será, além do conhecimento das melhores tecnologias utilizadas atualmente, também aquelas tecnologias que ainda estão incipientes, mas já se mostrando muito promissoras. Assim, seu estudo começará por conhecer e entender os tradicionais e já consolidados bancos de dados relacionais. Na sequência, em um mundo tão globalizado, é fundamental também adquirir conhecimentos em bancos de dados distribuídos, serviços de cloud e os bancos de dados NOSql (Big Data). Ainda, o estudo não poderia deixar de abordar alguns dos bancos de dados que, atualmente, estão emergindo como muito promissores. O que você acha? Acreditamos se tratar de temas muito estimulantes.

Vamos lá?

## Objetivos de aprendizagem

---

Ao final do estudo desta unidade, você será capaz de:

- conhecer bancos de dados relacionais e quando sua utilização é vantajosa;
- conhecer a implementação de um banco de dados relacional de forma distribuída;
- conhecer bancos de dados NoSql e suas melhores utilizações;
- conhecer os serviços de armazenamento em Cloud oferecidos pelo mercado;
- desenvolver capacidade analítica para a tomada de decisão na escolha de um banco de dados e a melhor forma de implementá-lo.



# 1.1 Bancos de dados

Você já ouviu falar em bancos de dados? É um conceito-chave em nossa trilha!

Uma definição clássica postula que um banco de dados é uma coleção de dados relacionados (ELMASRI; NAVATHE, 2018).

Sabemos que, qualquer que seja o objetivo de uma aplicação, o armazenamento do dado e a manipulação eficiente da informação são altamente relevantes. Sabemos também que, ao mesmo tempo em que o dado tem origem em diversas fontes, a informação emerge de todo um processo de seleção, organização e associação destes dados de maneira a trazer compreensão sobre o contexto desejado. É neste ponto que a tecnologia banco de dados nos fornece todas as ferramentas necessárias para a execução bem sucedida do objetivo



© whiteMocca / Shutterstock, 2021.

Veremos que a utilização de software e de metodologias apropriadas, aplicadas em conjunto, ajudam a determinar a melhor forma de relacionar, armazenar e disponibilizar a informação com o melhor desempenho. Conheceremos como isso funciona e, ainda, como utilizar esses softwares e metodologias conforme a nossa finalidade.

A seguir, veremos uma ferramenta importante quando se fala em banco de dados: O Sistema de Gerenciamento de Banco de dados (SGBD).

## 1.1.1 SGBD

---

O SGBD é um software que controla o armazenamento, recuperação, exclusão, segurança e integridade dos dados em um banco de dados. Segundo Elmasri e Navathe (2018, p. 5), ele “é um sistema de software de uso geral que facilita o processo de definição, construção, manipulação e compartilhamento de banco de dados entre diversos usuários e aplicações”.

Há vários tipos de SGBD e cada um deles foi projetado para apresentar melhor desempenho em requisitos específicos. Esta trilha terá um foco maior nos tipos Relacional e NoSQL. Estes tipos são os mais utilizados atualmente no mercado e endereçam necessidades específicas.

O modelo Relacional é projetado para tratar altos índices de I/O, ou seja, sistemas transacionais. Já o modelo NoSQL atende a necessidade de geração de informação analítica a partir de altos volumes de informação, como aqueles gerados, por exemplo, pela internet, e também em sistemas em que a consulta tem papel mais relevante que a manipulação dos dados.

Ao longo desta unidade, veremos que vários fatores devem ser analisados na decisão da utilização de um ou outro banco de dados. Além de critérios econômicos, ao longo do texto, serão tratados fatores técnicos que devem ser levados em consideração na seleção de um SGBD. Então, fique atento! Na maioria das vezes, os fatores a seguir estarão envolvidos nestas decisões.

- **Tipo de dado a ser armazenado:** Se estruturado, semiestruturado ou não estruturado.
- **Nível de concorrência a que o dado estará submetido:** Alto índice de I/O, processamento analítico com manipulação de altos volumes de informações ou baixa latência.
- **Escalabilidade:** A maior ou menor facilidade do aumento da capacidade de processamento.
- **Portabilidade:** Ligada à escalabilidade, a capacidade de ser utilizado em várias plataformas.

Em seguida, conheceremos os bancos de dados Relacionais. Acompanhe!

## 1.2 Bancos de dados Relacionais

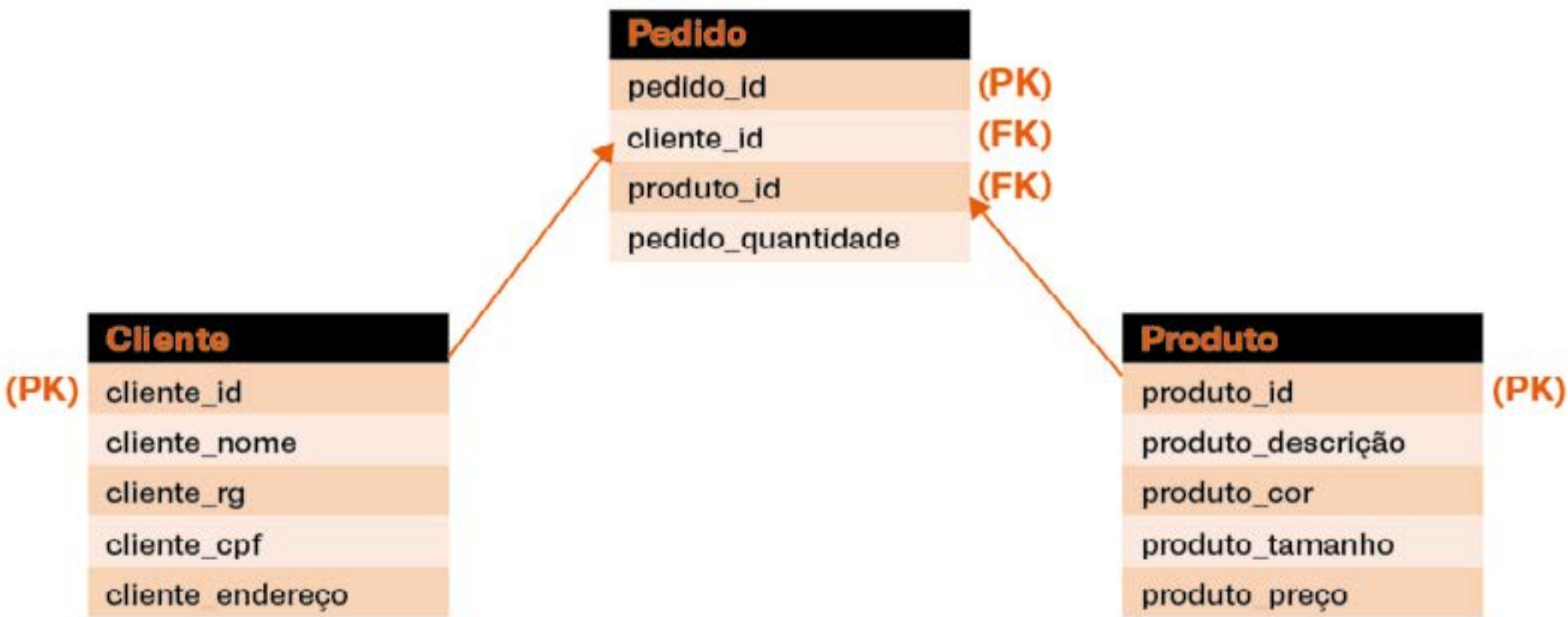
---

Um banco de dados relacional pode ser entendido como uma estrutura hierárquica de tabelas que se relacionam, e a maneira mais fácil de visualizar isso é fazendo uma correspondência com o Excel. Isso mesmo, o Excel!

Podemos imaginar as mesmas tabelas que montamos em planilhas do Excel, fazendo a seguinte correspondência: cada pasta Excel é uma tabela do banco de dados e cada tabela possui linhas e colunas. O relacionamento entre essas tabelas vai ocorrer através de colunas de ligação entre



elas. Exemplificando melhor, a tabela “A” contém uma coluna cujo conteúdo é chave para acessar a tabela “B”. Ou seja, o conteúdo da coluna de uma tabela (chave estrangeira) é vinculado à chave primária de outra tabela.



Fonte: elaborado pelo autor (2021).

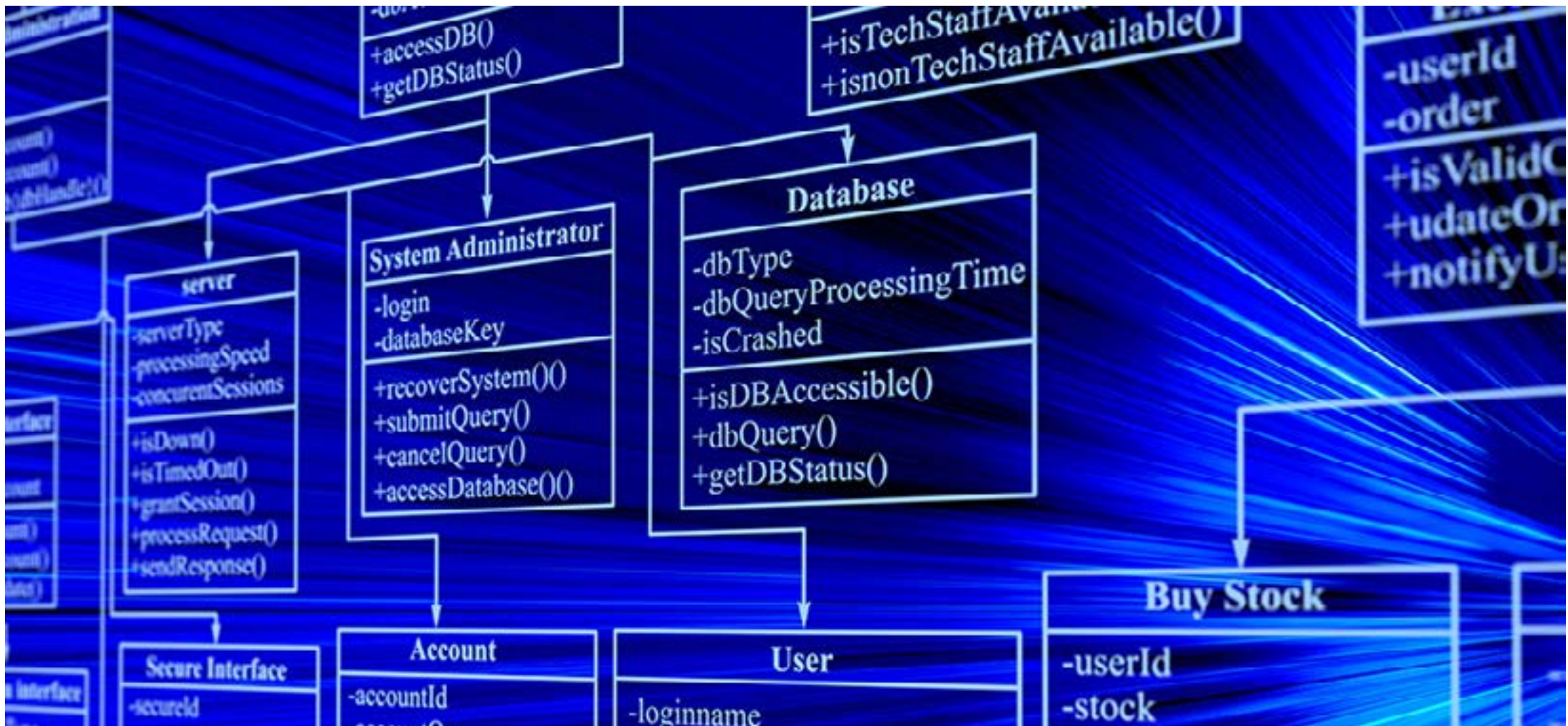
A figura acima demonstra um pequeno exemplo relacional. O **Pedido** se relaciona com o **Cliente** que efetuou a compra e com o **produto** comprado, e essas relações são concretizadas através das colunas **cliente\_id** e **produto\_id**. As tabelas **Pedido** e **Cliente** são relacionadas quando a coluna **cliente\_id** possui o mesmo conteúdo, ou seja, possuem o mesmo cliente. O mesmo ocorre entre Pedido e Produto, que são relacionados quando possuem o mesmo produto.

E você sabe quais as características do banco de dados Relacional? Confira a seguir.

## 1.2.1 Características de um Banco de dados Relacional

Como já foi dito anteriormente, um banco de dados relacional foi projetado para endereçar as necessidades de um alto volume de transações on-line. Nesses ambientes, a concorrência pelos dados é muito alta. Isso significa controlar, ao mesmo tempo, o armazenamento, recuperação, alteração, exclusão, segurança e integridade dessas transações. Significa também ter a habilidade de detectar, prontamente recuperar possíveis falhas e sincronizar o acesso aos dados compartilhados. Isso não é tarefa fácil. Controlar essas operações simultaneamente é bastante complexo! Para entender como isso é feito, primeiro você precisa entender o conceito de transação

em um banco de dados relacional.



Fonte: © Yermek / Shutterstock

Uma transação pode ser entendida como uma sequência de operações que inclui alguma manipulação (inserção, alteração ou exclusão) do dado em uma ou mais tabelas relacionadas. A sequência só faz sentido se todas as operações forem bem sucedidas. Executar apenas uma parte delas resultaria em um banco de dados inconsistente.

Assim, para que um SGBD possa oferecer a garantia de integridade dos dados em um ambiente de concorrência, é primordial que ele implemente as seguintes propriedades ACID.

**Atomicidade:** a transação deve ser concluída integralmente ou nenhuma das operações é concretizada. Por exemplo, em uma transação de saque da poupança e crédito do valor em conta corrente, as duas operações precisam ser bem sucedidas. A conclusão apenas do saque da poupança deixará o banco de dados inconsistente. A transação deve ser executada como uma unidade atômica.

**Consistência:** em caso de falha na execução da transação, o SGBD retorna todos os dados ao seu estado inicial, anterior ao início da transação, ou seja, consistente.

**Isolamento:** o SGBD deve garantir que transações concorrentes tenham uma prioridade de acesso e que uma não interfira no resultado da outra. Por exemplo, uma transação atualizando, e uma segunda consultando o mesmo dado. Nesse caso, o SGBD deve garantir um de dois



comportamentos possíveis — que a consulta seja feita após o dado ter sido atualizado, ou, então, que a consulta trabalhe com a cópia anterior à alteração ter sido feita.

**Durabilidade:** o SGBD deve garantir de que os dados não serão corrompidos ao longo do tempo, prezando por sua integridade.

Ao oferecer essas propriedades, o SGBD poupa nossas aplicações de terem que controlar todas essas situações. Ademais, em um ambiente extremamente requisitado, como aqueles com alto índice de I/O concorrente, o SGBD ainda nos dá alguma flexibilidade na manipulação dessas propriedades em benefício do desempenho. Teremos a oportunidade de conhecer isso, em maior detalhe, mais adiante.

Além destas propriedades, de acordo com Elmasri e Navathe (2018), um SGBD Relacional ainda possui algumas características que facilitam ao desenvolvedor e que também o diferem de outros sistemas de arquivos. Veja a seguir quais são essas características.

**Autodescrição:** o SGBD mantém um catálogo com as informações do banco de dados, e esse catálogo é detalhado a ponto de “autodescrever” a arquitetura do banco. Informações como nome; tipo do dado, índices, restrições de integridade, são mantidas na forma de metadados e oferecem muita facilidade na compreensão dos dados armazenados e suas relações.

**Isolamento entre programas e dados:** a estrutura de dados não mantém vínculo com a aplicação e isso nos permite alterá-la sem ter que também alterar a aplicação.

**Suporte a múltiplas visões dos dados:** a partir da estrutura original, é possível criar uma visão, ou seja, um subconjunto do banco de dados, contendo apenas o dado de interesse do usuário. Este subconjunto pode, inclusive, conter dados virtuais, derivados de um cálculo, por exemplo. Dessa forma, o usuário não precisa conhecer ou entender toda estrutura de dados, apenas os dados de seu interesse.

**Segurança em processamento multiusuário e compartilhamento dos dados:** você já deve ter percebido, ao conhecer as propriedades ACID, o SGBD precisa ter pleno domínio das transações concorrentes e, com isso, manter os dados consistentes.

Vamos conferir, agora, quando utilizar um banco de dados relacional?

## 1.2.2 Quando usar um banco de dados relacional

Até aqui, observando as propriedades ACID e as características de um banco de dados relacional, já poderemos enumerar várias características que podem ser vantajosas no uso de um banco de dados relacional. Controle de concorrência, garantia da integridade dos dados, facilidade de manutenção da estrutura de dados com um mínimo impacto na aplicação e processamento multiusuário. São vantagens que levaram essa arquitetura de banco de dados a dominar amplamente o mercado de aplicações intensivas em I/O.

Entretanto, o “ecossistema” relacional ainda nos traz outra vantagem. Neste ambiente, a técnica de modelagem dos dados gera um modelo claro dos relacionamentos e impõe redundância controlada. Além de permitir melhor desempenho na recuperação do dado em ambientes intensivos em I/O concorrentes, isso também reduz a complexidade e o tempo de desenvolvimento de aplicações. Mas não se preocupe, isso será detalhado na Unidade 3.

Há ainda outro tipo de aplicação que se beneficia muito das características apresentadas: as Aplicações OLAP (On-line Analytical Processing). Aplicações OLAP atendem a necessidade de análise de dados corporativos com objetivo de tomada de decisão estratégica, comumente chamado Business Intelligence (BI).



Estes dados são obtidos a partir de um sistema corporativo e são consolidados e armazenados em uma estrutura modelada especialmente para proporcionar várias visões do negócio, por exemplo, produtos mais vendidos, previsões de crescimento e média de vendas por cliente.

O armazenamento desses dados, consolidados em um banco de dados especialmente modelado para este fim, é chamado Data Warehouse. Os dados são obtidos a partir de uma fonte estruturada e consistente, e um banco de dados Relacional atende bem essa necessidade. De fato, as aplicações de Business Intelligence são modeladas em banco de dados relacional.

Podemos citar vários exemplos de bancos de dados relacionais muito atuantes no mercado. Recomendamos o acesso a alguns dos melhores fornecedores, conforme podemos ver a seguir.

**Oracle:** <https://www.oracle.com/br/database/>

**SQL Server:** <https://www.microsoft.com/pt-br/sql-server/sql-server-downloads>

**Postgresql:** <https://www.postgresql.org/>

**MySQL:** <https://www.mysql.com/>

Ao acessá-los, você terá oportunidade de conhecer, examinar e refletir sobre as características e diferenciais de cada um dos produtos.

## 1.3 Arquitetura de um SGBD Relacional

Até agora, conhecemos **O QUE** um **SGBD** Relacional faz, mas é importante também entender **COMO** se faz tudo isso. Um SGBD é composto por módulos que interagem entre si no objetivo de entregar a tarefa solicitada. Neste tópico, compreenderemos como se processam algumas atividades que rotineiramente um desenvolvedor terá contato no desenvolvimento de suas aplicações. Entre as ações estão as listadas a seguir.

**Catálogo do banco de dados:** armazena informações que identificam e quantificam os conteúdos do banco de dados. Informações como nome das tabelas e colunas, tipos de dados, chaves de acesso, índices, restrições e várias informações estatísticas, por exemplo, o volume dos dados em cada tabela.

**Compilador:** os comandos de manipulação dos dados são escritos em uma linguagem de programação, no caso do SGBD relacional, o SQL. Ao receber essas instruções da aplicação, o compilador valida a sintaxe do comando, interpreta e gera o comando de acesso que deve ser encaminhado ao Runtime.

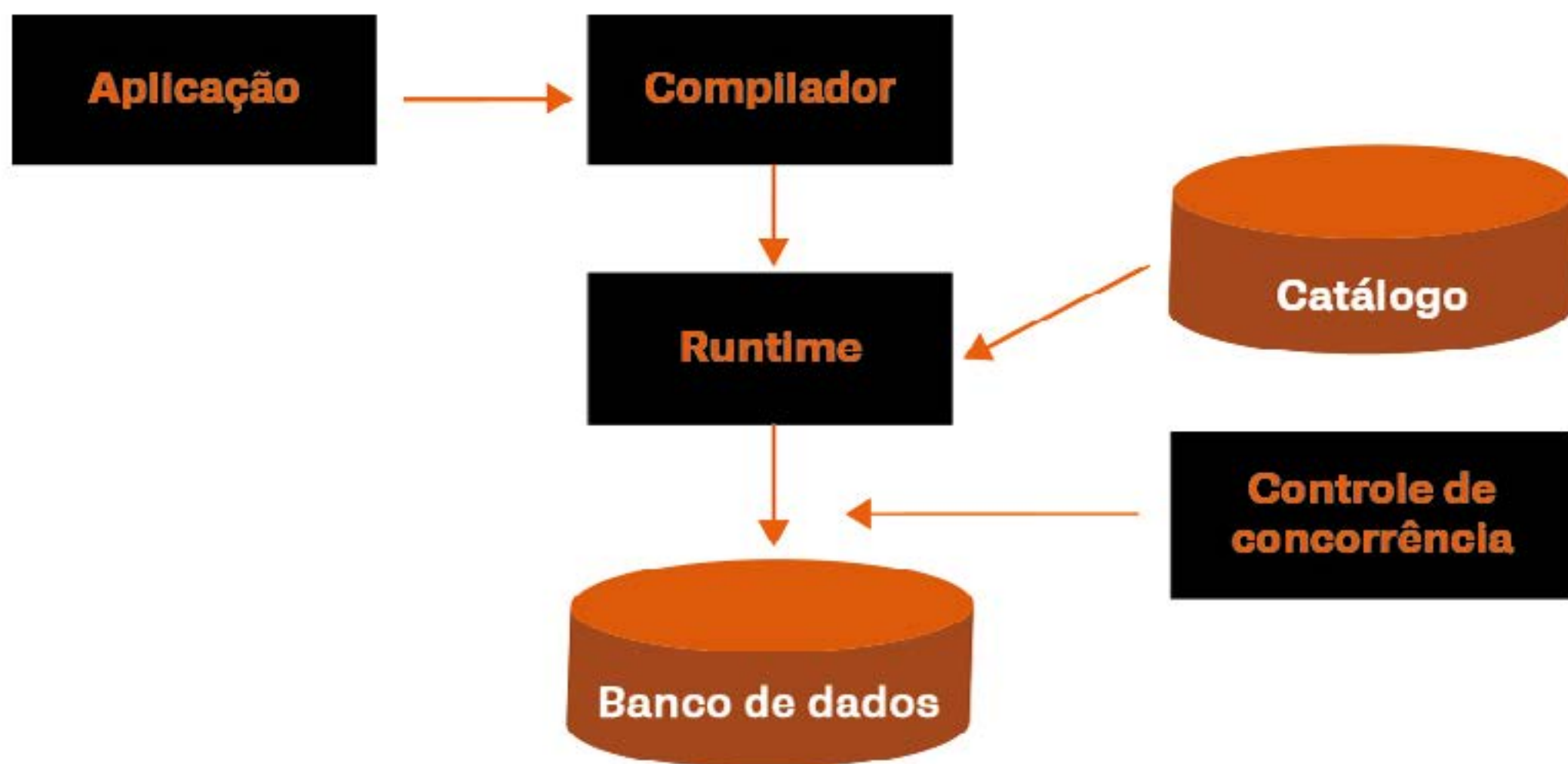


**Runtime:** controla e determina a melhor estratégia de acesso aos dados em tempo de execução. Neste caso, é importante notar o que significa “em tempo de execução”. A melhor estratégia de acesso é determinada dinamicamente com base no estado atual dos índices e informações do catálogo do banco de dados. Isso significa que essa determinação dinâmica pode revelar, “de uma hora para outra”, uma queda no desempenho, devido, por exemplo, a altos volumes ou à necessidade da implementação de um novo índice. Essa melhor estratégia de acesso é chamada Plano de Execução, e é armazenada em uma tabela do catálogo do banco de dados, possibilitando consulta pelo DBA ou pelo desenvolvedor. Contém informações muito úteis para melhoria do desempenho da aplicação.

**Controle de concorrência:** em um ambiente de acesso concorrente, este módulo é responsável por manter os dados disponíveis e íntegros. Com base nos parâmetros do banco de dados e da aplicação, em caso de falha, é ele quem decide qual transação tem prioridade e deve ser efetivada. Na Unidade 4, voltaremos a esse assunto focando principalmente na importância disso para o desempenho geral do sistema.

**Gerenciamento dos dados armazenados:** é responsável por armazenar os dados conforme as informações do catálogo de dados.

A figura a seguir representa a interação entre estes módulos.



O SGBD também oferece alguns utilitários importantes na a rotina de sua manutenção e gerenciamento. É importante conhecê-los. Veja a seguir.

## 1.3.1 Utilitários do SGBD

---

O SGBD também oferece utilitários que facilitam a rotina de gerenciamento, administração e análise do banco de dados. Vamos conhecê-lo na sequência.

**Loading:** este utilitário executa uma carga maciça de dados nas tabelas. É utilizado nos casos em que a alimentação inicial do banco de dados não acontece através da aplicação e a fonte dos dados é externa.

**Backup:** gera uma cópia de segurança de todas as informações do banco de dados. Tabelas, índices, catálogo, ou seja, todas as informações necessárias para uma possível restauração do banco de dados. Esse utilitário também oferece a opção de realização de backup incremental, onde apenas as modificações ocorridas a partir de um backup completo serão salvas.

**Reorganização do banco de dados:** executa a reconstrução dos índices das tabelas eliminando possíveis fragmentações nessas estruturas. Após a reconstrução, todo o índice estará organizado e contido em páginas contínuas de disco, proporcionando melhor desempenho nos acessos ao banco de dados.

**Monitoramento das estatísticas:** expõe as informações de desempenho do banco de dados em forma de estatísticas. Além de informações importantes para um administrador de banco de dados, oferece também informações que geram oportunidades de melhorias nas aplicações, por exemplo, consultas que estão exigindo alto consumo de recursos do banco de dados.

## 1.3.2 Opções de implementação de um banco de dados relacional

---

Existem algumas formas de se implementar um banco de dados e isso vai depender, principalmente, da necessidade do usuário e do hardware disponível para essa implementação. De maneira geral, as implementações Cliente/Servidor, Distribuída e a utilização de serviços na nuvem (Cloud) atendem a maioria das necessidades, veja abaixo.

**Ambiente cliente/servidor:** aplicação é executada em uma estação de trabalho (cliente) e o banco de dados é instalado em um hardware dedicado ao SGBD (servidor). A conexão entre o cliente e o



servidor é feita através de rede. Dessa maneira, é possível conectar várias estações de trabalho ao mesmo banco de dados executada pelo fornecedor. Toda essa complexidade estará transparente ao cliente do serviço e o acesso a seus dados serão disponibilizados através da nuvem.

Tivemos uma breve explicação sobre uma implementação distribuída de banco de dados. Este é um tipo de implementação muito utilizado, e vale a pena se aprofundar mais nisso. A seguir,

## 1.4 Bancos de dados distribuídos

Em um mundo globalizado, é comum que as empresas estejam distribuídas geograficamente e os custos de se manter um banco de dados centralizado pode não ser vantajoso. Em uma implementação distribuída, pode-se conseguir ganhos financeiros e operacionais ao negócio. Os ganhos financeiros podem estar no crescimento modular do hardware e diminuição nos custos de telecomunicações, e o ganhos operacionais, na proximidade do dado ao usuário utilizador e seu desenvolvedor, por exemplo.





Além disso, uma falha qualquer no ambiente (comunicação, hardware, software) afetará apenas um fragmento e não a totalidade do banco de dados.

Agora, chegou o momento de conferir as técnicas de armazenamento de bancos de dados distribuídos. Vamos lá?

## 1.4.1 Técnicas de armazenamento de bancos de dados distribuídos

Neste contexto, surge a seguinte pergunta: como ocorre concretamente o armazenamento de um banco de dados distribuído?

Existem três estratégias, conforme vemos abaixo.

**Replicação:** o SGBD mantém réplicas idênticas do dado em localidades diferentes. Essa abordagem permite um melhor desempenho nas operações de leitura, porém causa maior sobrecarga nas operações de atualização. Essa replicação dos dados ainda pode ser feita de forma síncrona e assíncrona. Na forma síncrona, todas as cópias do dado são feitas no momento da replicação, e uma alteração no dado, demanda a atualização sincronizada de todas as cópias. Na forma assíncrona, as cópias poderão ficar fora de sincronia. Quando a alteração é feita, a replicação é agendada para um outro momento, a depender da necessidade da aplicação.

**Fragmentação:** os dados são fragmentados de acordo com uma regra lógica, e isso pode ser feito de forma horizontal ou vertical. Na forma horizontal, a regra é aplicada às linhas de uma tabela, fragmentando, por exemplo, todos os clientes que correspondem àquela localidade geográfica. Dessa maneira a tabela de clientes estaria fragmentada por tantas quantas fossem as localidades de interesse do negócio. Na forma vertical, a regra é aplicada às colunas da tabela. Uma aplicação dessa forma de fragmentação, poderia ser o caso em um departamento que cuida da atualização de apenas uma parte das informações de uma entidade. Por exemplo, um cadastro de clientes mantido pelo departamento de vendas, porém, o limite de crédito desses clientes é mantido pelo departamento financeiro em uma localidade diferente.

**Replicação e Fragmentação:** é a utilização conjunta da fragmentação e da replicação de dados.

E aí, compreendeu quais as principais técnicas utilizadas e com qual finalidade? A seguir, veremos quais técnicas de armazenamento distribuído implicam em maior complexidade no controle de concorrência e quais pontos devemos dar atenção em implementações como essas.

## 1.4.2 Questões para o controle de concorrência em ambientes distribuídos

---

Muitas situações que não acontecem em um ambiente centralizado podem ocorrer em um ambiente distribuído. Com o que sabemos até agora, você poderia mencionar algumas destas situações? Vamos ver algumas delas a seguir.

**Múltiplas cópias dos dados:** o controle de concorrência deve manter a consistência de cada fragmento, e o dado deve ser consistente entre todas as suas réplicas.

**Falhas em nós:** uma das vantagens de um ambiente distribuído é a disponibilidade em caso de falhas. Em contrapartida, a recuperação dessas falhas exige que o banco de dados deva ser atualizado em relação a todos os outros fragmentos.

**Falhas na rede:** com base na arquitetura de distribuição implementada, o sistema deve ter habilidade para lidar com as várias combinações de falhas de comunicação em um ou mais sites.

**Deadlock distribuído:** um deadlock ocorre quando há um impasse na tentativa de atualização no banco de dados, por exemplo, quando duas transações se colocam em espera uma da outra em um loop infinito. Em um ambiente distribuído, as técnicas para lidar com essa situação devem ser estendidas para considerar a fragmentação e a replicação dos dados.

Como você pode perceber, é importante que estas questões sejam tratadas adequadamente. Veremos, no próximo tópico, que o Teorema equaciona todas estas questões.

## 1.4.3 Teorema CAP

---

Você conseguiu perceber que o controle de concorrência em sistemas distribuídos é muito mais complexo do que em um ambiente centralizado? Conhecemos, até aqui, algumas falhas que podem ocorrer nestes ambientes. Sabemos também que o banco de dados deve garantir as propriedades **ACID** (**a**tomicidade, **c**onsistência, **i**solamento e **d**urabilidade) para garantir um dado íntegro em um ambiente concorrente.

Então, como é feita a garantia de integridade em sistemas distribuídos? Segundo Elmasri e Navathe (2018), o Teorema CAP foi elaborado para equacionar todas essas questões. Ao conhecer o teorema, você vai entender esses comportamentos.

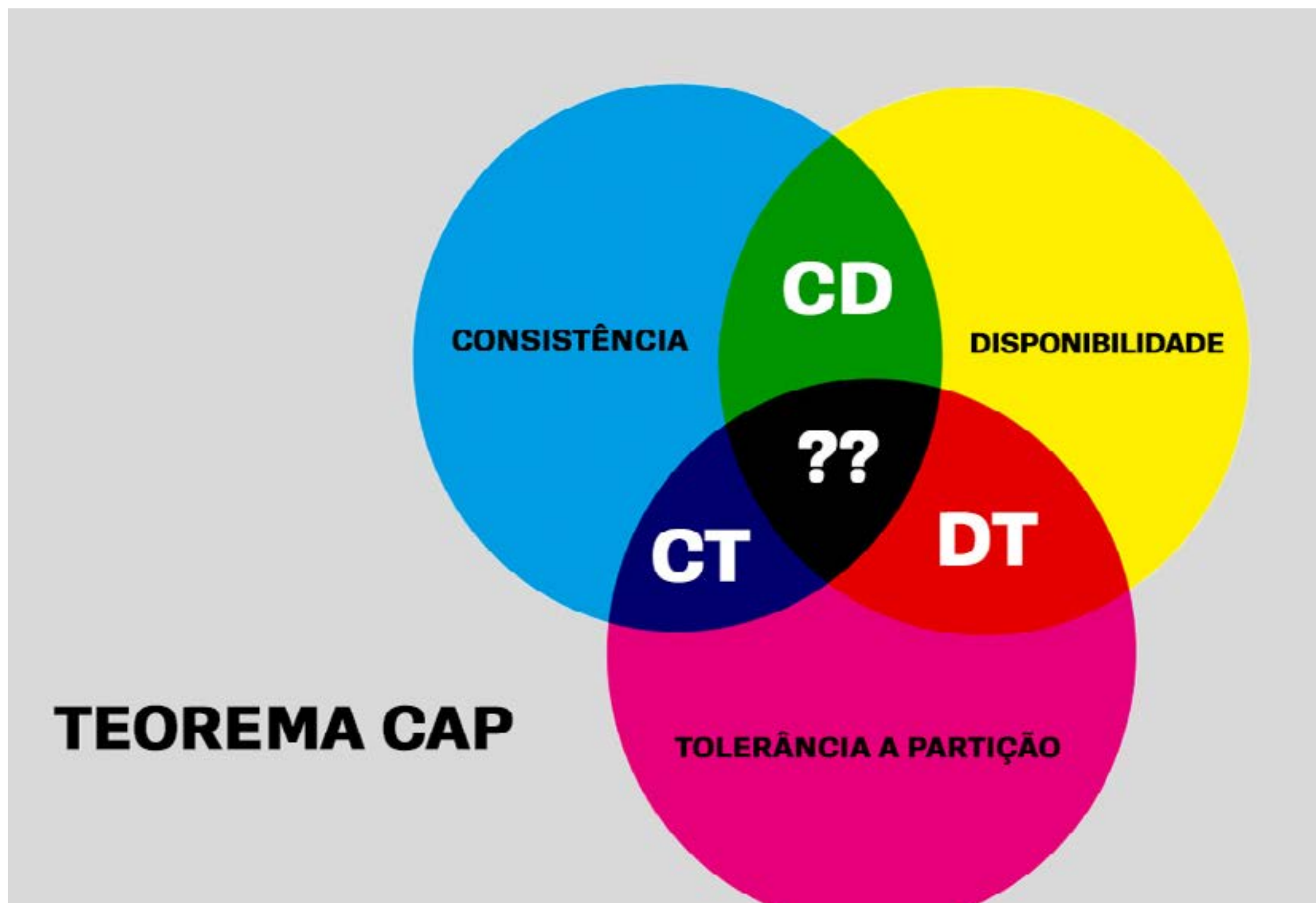
Esse teorema expõe os seguintes comportamentos possíveis de se adotar em um banco de dados distribuído quando ocorrem operações concorrentes de escrita e leitura.

**Consistente (Consistent):** é garantido o retorno do dado mais atualizado independentemente de qual site seja consultado.

**Disponível (Available):** sempre será retornado um valor, desde que ao menos um servidor esteja disponível. Decorre que esse dado pode estar desatualizado.

**Tolerante à partição (Partition Tolerant):** se houver falha na comunicação em algum dos nós, os demais poderão responder às solicitações de consulta.

Observe esta dinâmica na figura a seguir.



Fonte: elaborado pelo autor (2021).

O teorema ainda deduz que é possível garantir apenas a combinação de dois desses comportamentos: Disponível e Consistente; Disponível e Tolerante à Partição; Consistente e Tolerante à Partição. Não é possível garantir os três comportamentos ao mesmo tempo. Como um bom exercício mental, reflita: Por quê?



Vamos refletir juntos: A chave para entender por que não é possível ter os três comportamentos ao mesmo tempo, é refletir com base na disponibilidade e na tolerância à partição. Quando um banco de dados for Tolerante à Partição, já se assume a possibilidade de falhas, e, dessa forma, teremos que escolher se o banco de dados será Consistente ou Disponível. Estas opções são mutuamente excludentes. Se escolhermos disponibilidade, sabemos que poderá haver alguma inconsistência. Se escolhermos consistência, sabemos que não será possível o banco de dados estar 100% disponível. Resumindo, um banco de dados tolerante à partição, ou será consistente, ou será disponível, não é possível um banco tolerante à partição que seja ao mesmo tempo disponível e consistente. Por outro lado, quando o requisito principal for disponibilidade, devemos escolher entre consistência e tolerância à partição.

Entretanto, toda esta complexidade de implementação pode ficar transparente quando se contrata serviços de um fornecedor especializado em banco de dados. No próximo tópico, veremos como a utilização de um banco de dados na nuvem pode ser vantajosa para o cliente e para o desenvolvedor.

## 1.5 Cloud – DbaaS: Banco de dados como um serviço

---

Vimos sobre a possibilidade de implementação de um banco de dados na nuvem utilizando os serviços de um fornecedor e, neste tópico, você conhecerá mais sobre isso.

O conceito de **computação em nuvem** é intimamente ligado a um modelo de fornecimento de serviços de TI que permite ao cliente focar nas atividades estratégicas de seu próprio negócio. O fornecedor possui todos os recursos materiais e toda expertise no serviço contratado, e coloca isso a serviço do cliente. O cliente pode, então, focar em sua aplicação e requisitos de negócio, sem se preocupar com a complexidade de todos os recursos envolvidos na manutenção de um banco de dados. Questões como inovação, especialistas e escalabilidade do hardware deixam de estar no escopo de preocupações do cliente. Esse modelo de negócios de TI está disponível também para infraestrutura, plataforma e software. Vejamos a seguir.

**Infraestrutura (IaaS – Infrastructure as a Service):** nesta modalidade, o cliente contrata o hardware para utilização, como capacidade de memória e processamento. O cliente paga pela utilização, e o controle sobre o hardware contratado é executado pelo cliente.

**Plataforma (PaaS – Platform as a Service):** o cliente contrata o acesso a uma plataforma de desenvolvimento. É uma alternativa muito utilizada por empresas desenvolvedoras de software.

**Software (SaaS – Software as a Service):** o cliente contrata a disponibilização remota de um

software. Este modelo diminui sobremaneira os custos com instalação e licenças do software. Além disso, obtêm-se ganhos de elasticidade (aumentar ou reduzir sem dificuldade), mobilidade e atualização tecnológica.

**Banco de dados (BDaaS – Bata Base as a Service):** o cliente contrata todo o ambiente para a utilização do banco de dados disponibilizado pelo fornecedor, acessando estes serviços através da internet.

Existem vários fornecedores de serviços em cloud e recomendamos que aperfeiçoe seus conhecimentos iniciando os estudos pelos fornecedores listados abaixo

**Microsoft:** <https://azure.microsoft.com/pt-br/>

**Amazon:** <https://aws.amazon.com/pt/>

**Google:** <https://cloud.google.com/>

## 1.6 Bancos de dados NoSQL (Not Only Sql)

---

Vimos sobre a possibilidade de implementação de um banco de dados na nuvem utilizando os serviços de um fornecedor e, neste tópico, você conhecerá mais sobre isso.

O conceito de **computação em nuvem** é intimamente ligado a um modelo de fornecimento de serviços de TI que permite ao cliente focar nas atividades estratégicas de seu próprio negócio. O fornecedor possui todos os recursos materiais e toda expertise no serviço contratado, e coloca isso a serviço do cliente. O cliente pode, então, focar em sua aplicação e requisitos de negócio, sem se preocupar com a complexidade de todos os recursos envolvidos na manutenção de um banco de dados. Questões como inovação, especialistas e escalabilidade do hardware deixam de estar no escopo de preocupações do cliente. Esse modelo de negócios de TI está disponível também para infraestrutura, plataforma e software. Vejamos a seguir.

Alta escalabilidade	Embora os bancos de dados relacionais também sejam escaláveis, seja por aumento no poder de processamento ou na forma distribuída, a complexidade do controle de concorrência aumenta consideravelmente. Em bancos de dados NoSql, adicionar armazenamento e processamento de forma distribuída não adiciona complexidade ao banco de dados.
Consistência eventual	Muitas aplicações NoSql permitem que se trabalhe de forma mais flexível em relação à consistência. Nesses casos, a aplicação pode ser planejada de maneira a tolerar leves inconsistências no banco de dados aumentando muito o desempenho dessas aplicações.
Alta disponibilidade	Com suporte nativo à replicação e à consistência eventual, a disponibilidade aumenta consideravelmente.
Ausência de esquema	Trabalhar com dados não estruturados pode exigir ausência de esquema. Pode-se trabalhar com esquemas parciais, porém, não é obrigatório. Uma consequência disso é que não há garantia de integridade do dado como ocorre no modelo relacional. Porém, muitas aplicações analíticas em grandes volumes de dados não requerem esta integridade tão rígida.

Diferentemente dos bancos de dados relacionais, existem vários tipos de bancos de dados NoSql, cada um deles endereça em necessidades específicas. No próximo tópico, veremos os tipos mais utilizados.

Existem algumas listas na internet dedicadas a bancos de dados NoSql. [Neste link](#), você poderá ver uma lista de servidores NoSql categorizada pelo tipo de armazenamento. Já [neste link](#), você encontrará informações a considerar na escolha de um banco de dados NoSql.



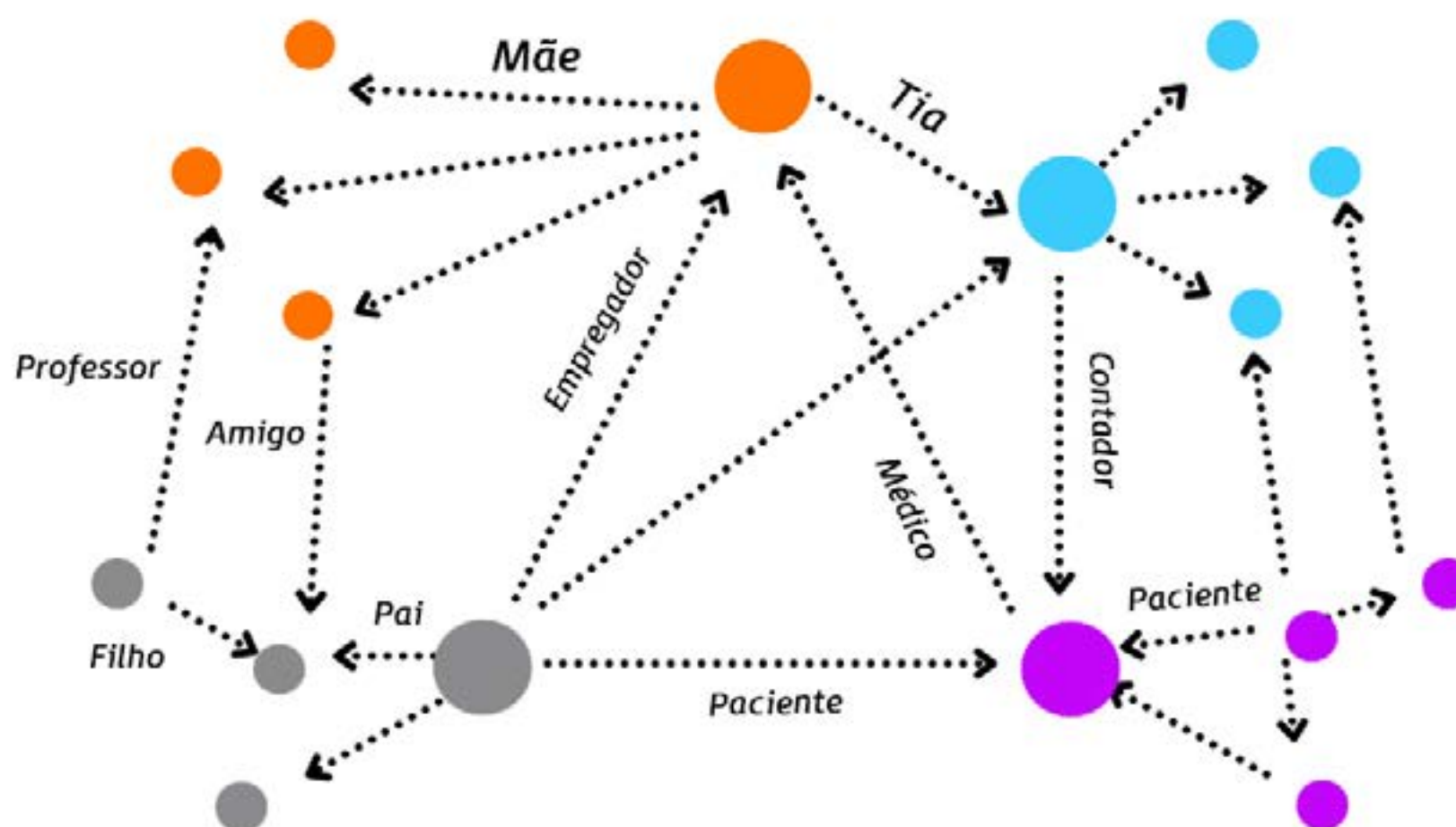
## 1.6.1 Tipos de bancos de dados NoSql

Em função da variedade de fontes de dados não estruturados, ao longo do tempo, foram surgindo bancos NoSql otimizados para o armazenamento e acesso especializados ao tipo de dado armazenado: Podemos citar 4 principais tipos que veremos nos próximos subtópicos. Acompanhe!

### 1.6.1.1 Orientados a documentos

Nestes bancos, os dados são armazenados na forma de documentos autoexplicativos, isto é, o documento já define a estrutura e o significado dos dados que ele contém. Podemos citar como exemplo um documento JSON (JavaScript Object Notation). JSON é um padrão de documento para intercâmbio de informações entre sistemas e transmite a informação em formato texto. É similar ao padrão XML.

Uma aplicação para este tipo de banco de dados é o gerenciamento de conteúdo e blogs, e alguns bancos que suportam esse padrão são o MondoDB, CouchDB e o RavenDB.

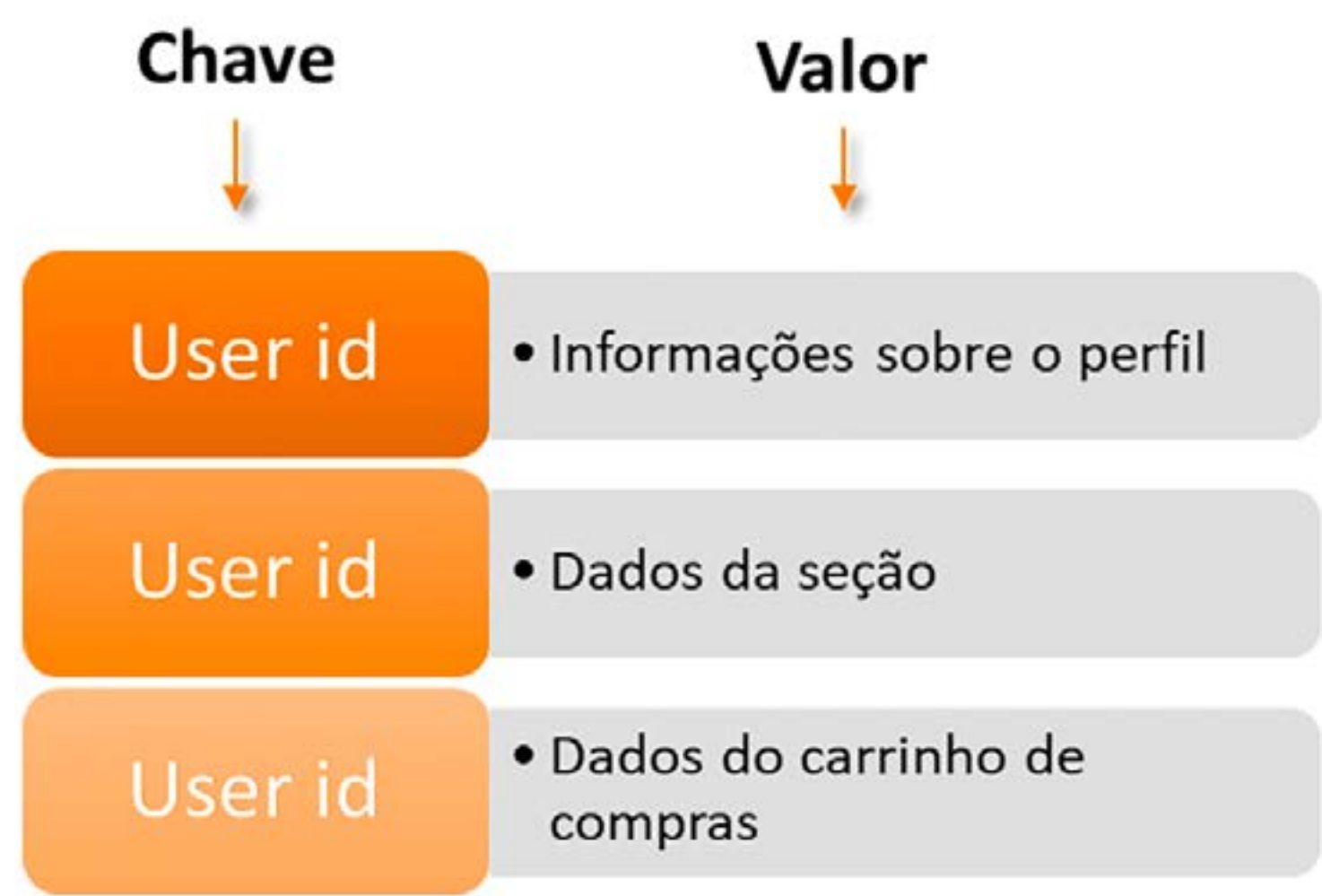


Fonte: elaborado pelo autor (2021).

Observando o documento acima, percebemos que ele é livre de esquema e permite redundância e inconsistência.

# 1.6.1.2 Chave-valor

Estes bancos de dados trabalham com um conjunto de pares chave-valor, sendo a chave um identificador único. Esse tipo de banco de dados é muito utilizado para armazenamento de dados a respeito de uma seção de navegação na web, como o perfil do usuário e o carrinho de compras. Ao acessar essas informações, não são necessários relacionamentos ou uso de múltiplas chaves. Veja a seguir uma representação de um banco de dados do tipo chave-valor



Fonte: elaborado pelo autor (2021).

E observe, a seguir, a configuração de uma armazenagem em multi colunas.



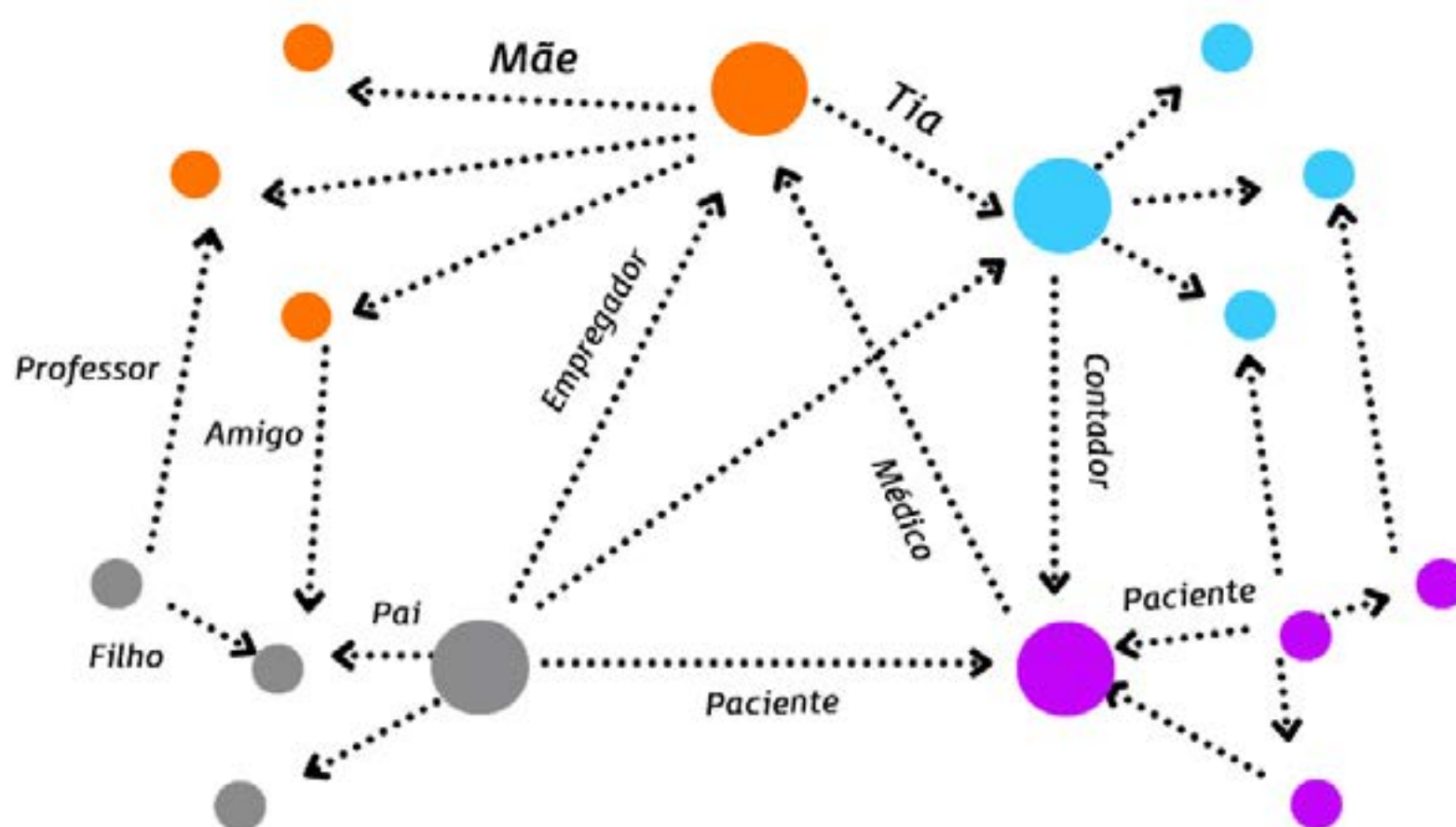
Fonte: elaborado pelo autor (2021).

O Cassandra (Facebook) e o HBase são exemplos de modelo de armazenagem multi colunas.

Os dados podem, ainda, serem armazenados como grafos, é o que veremos a seguir.

## 1.6.1.4 Baseados em grafos

Os dados são armazenados conceitualmente como grafos com nós relacionados, e a leitura percorre suas arestas por meio de expressões que determinam o caminho a seguir. Atendem à necessidade de aplicações como redes sociais e sistemas de recomendação.



Fonte: elaborado pelo autor (2021).

Os dados são armazenados conceitualmente como grafos com nós relacionados, e a leitura percorre suas arestas por meio de expressões que determinam o caminho a seguir. Atendem à necessidade de aplicações como redes sociais e sistemas de recomendação.

## 1.6.2 Vantagens de um banco de dados NoSql

Os dados são armazenados conceitualmente como grafos com nós relacionados, e a leitura percorre suas arestas por meio de expressões que determinam o caminho a seguir. Atendem à



necessidade de aplicações como redes sociais e sistemas de recomendação.

**Quer entender um pouco mais sobre o cenário de mudança tecnológica em bancos de dados? [Acesse este artigo](#) da revista ComputerWorld a respeito do tema, publicado em 21/01/2021. O autor descreve uma evolução em curso na utilização de bancos de dados pelo setor financeiro**

Conheceremos, na sequência, alguns bancos de dados que têm se mostrando bastante promissores.

## 1.7 Bancos de dados promissores

Sabemos que o mundo de hoje é movido a inovações, e a tecnologia com certeza é uma ferramenta de geração e suporte à inovação. Sendo assim, é muito importante que você acompanhe as tendências tecnológicas e a movimentação no mercado de TI. Vamos conhecer os bancos de dados que têm se mostrado promissores? Iniciaremos pelo DynamoDB.

### 1.7.1 Amazon Web Services: DynamoDB

O DynamoDB é um produto da Amazon e faz parte da plataforma Amazon Web Services. O serviço inclui gestão automática do banco de dados com backups contínuos, replicação, ferramentas de monitoração e escalabilidade elástica e automática.

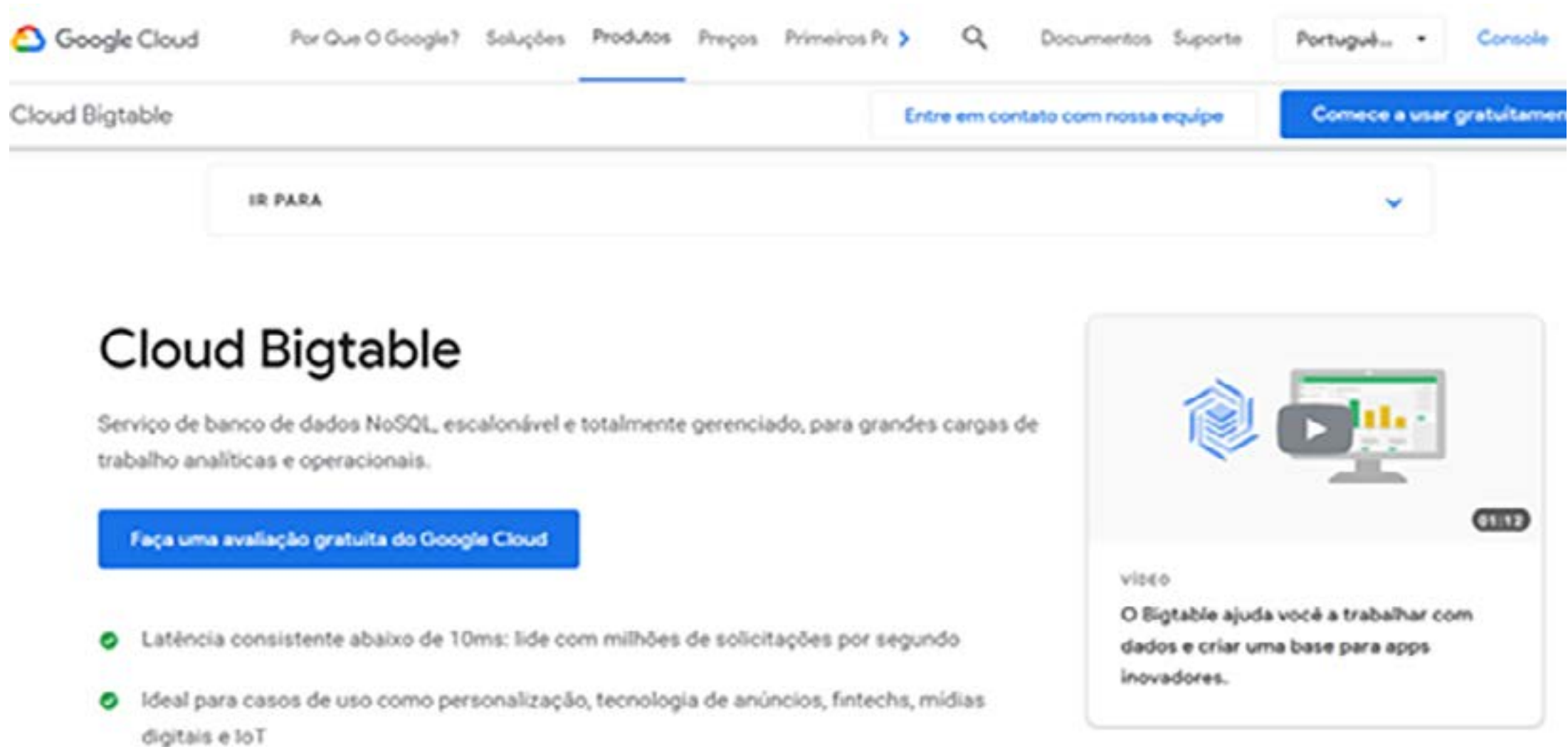


Fonte: Amazon DynamoDB | Banco de dados chave-valor NoSQL | Amazon Web Services

O DynamoDB oferece ainda uma API de desenvolvimento simples e funcional. O valor do serviço é determinado em função da capacidade mínima e máxima desejada, utilização do serviço e da região. Muito rápido e flexível, é muito utilizado em web, dispositivos móveis e aplicações de baixa latência, como jogos.

## 1.7.2 Google: BigQuery e BigTable

BigTable e BigQuery fazem parte do pacote de soluções em nuvem da Google. BigTable é um sistema de armazenamento distribuído, proprietário do Google, que trabalha com dados estruturados, porém projetado para lidar com petabytes de dados em milhares de servidores. Não é um banco de dados relacional, então não oferece suporte a consultas com junções ou suporte ao SQL. Entretanto, apresenta altíssimo desempenho em grande escala de dados. Serviços como Google Maps e a indexação da web têm seus dados armazenados neste banco de dados.



Google Cloud

Por Que O Google? Soluções Produtos Preços Primeiros Passos >

Documentos Suporte

Português

Console

Cloud Bigtable

Entre em contato com nossa equipe

Comece a usar gratuitamente

IR PARA

### Cloud Bigtable

Serviço de banco de dados NoSQL, escalonável e totalmente gerenciado, para grandes cargas de trabalho analíticas e operacionais.

Faça uma avaliação gratuita do Google Cloud

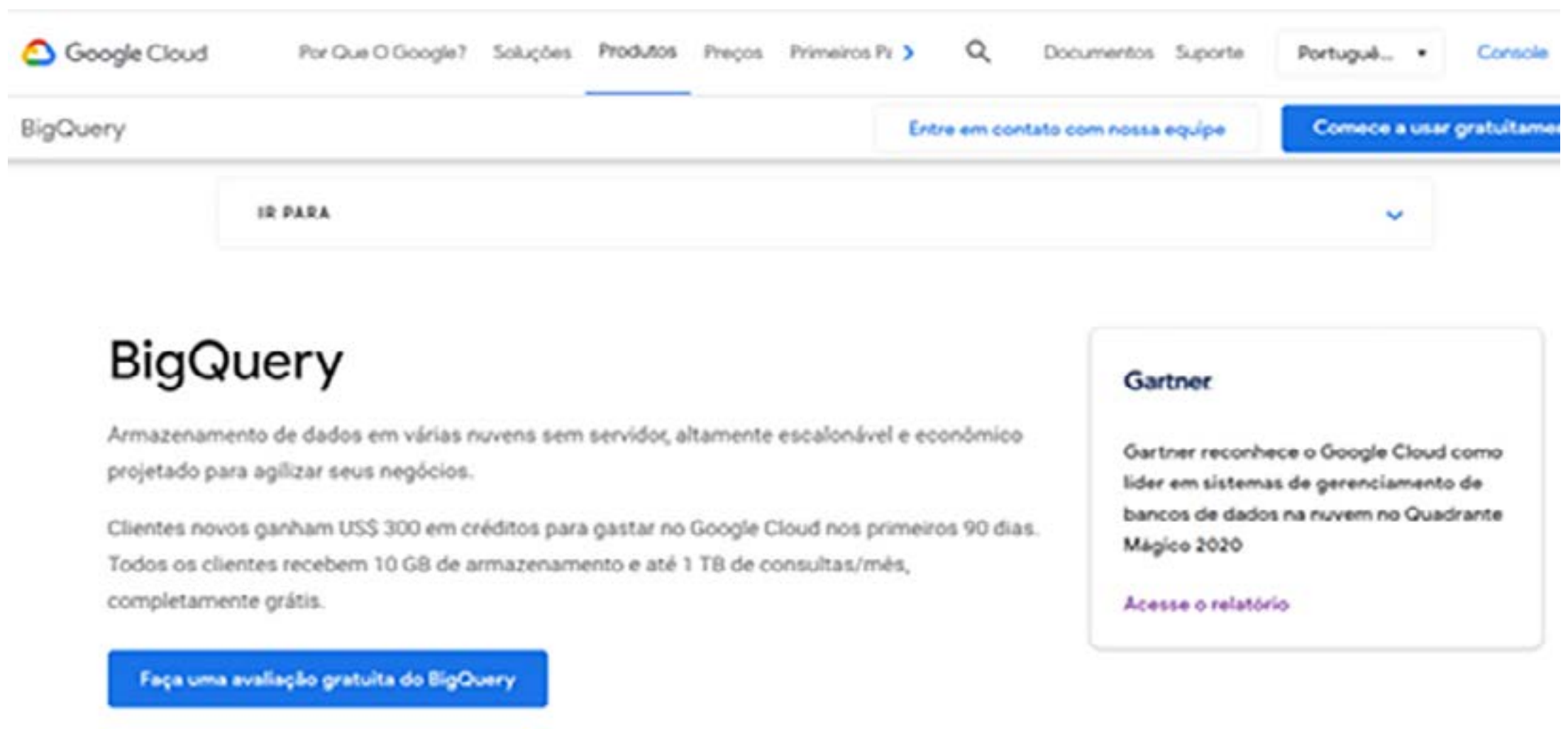
- ✓ Latência consistente abaixo de 10ms: lide com milhões de solicitações por segundo
- ✓ Ideal para casos de uso como personalização, tecnologia de anúncios, fintechs, mídias digitais e IoT

video

O Bigtable ajuda você a trabalhar com dados e criar uma base para apps inovadores.

Fonte: Cloud Bigtable: serviço de banco de dados NoSQL | Google Cloud

Já o BigQuery é um modelo de serviço com características voltadas a data warehouse.

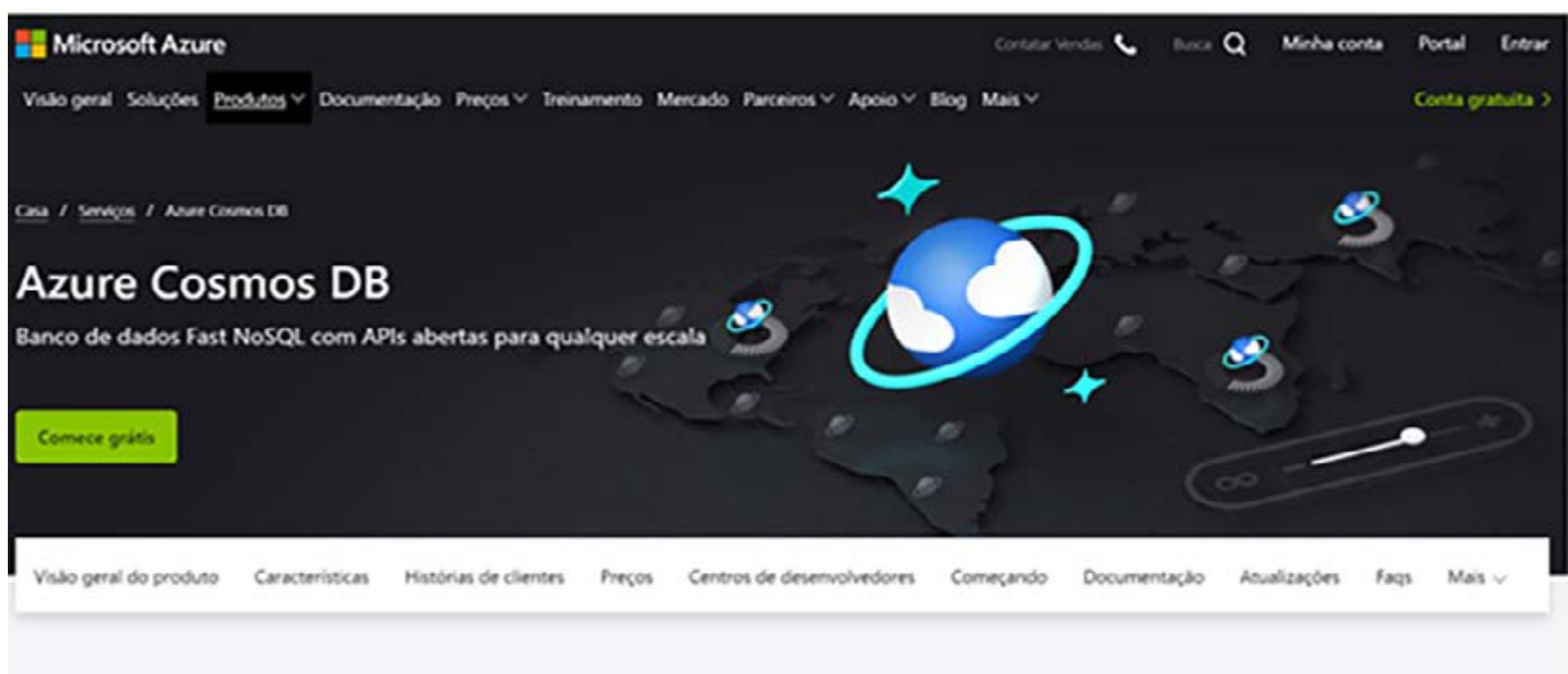


Fonte: BigQuery: armazenamento de dados na nuvem | Google Cloud

Diferentemente do BigTable, esta ferramenta oferece suporte ao SQL através de uma interface Web e a ferramenta pode interagir com outros serviços do Google.

## 1.7.3 Microsoft: Cosmos DB

Cosmos DB é o banco de dados da plataforma de cloud Azure da Microsoft.



Fonte: Azure Cosmos DB – Non-Relational Database | Microsoft Azure

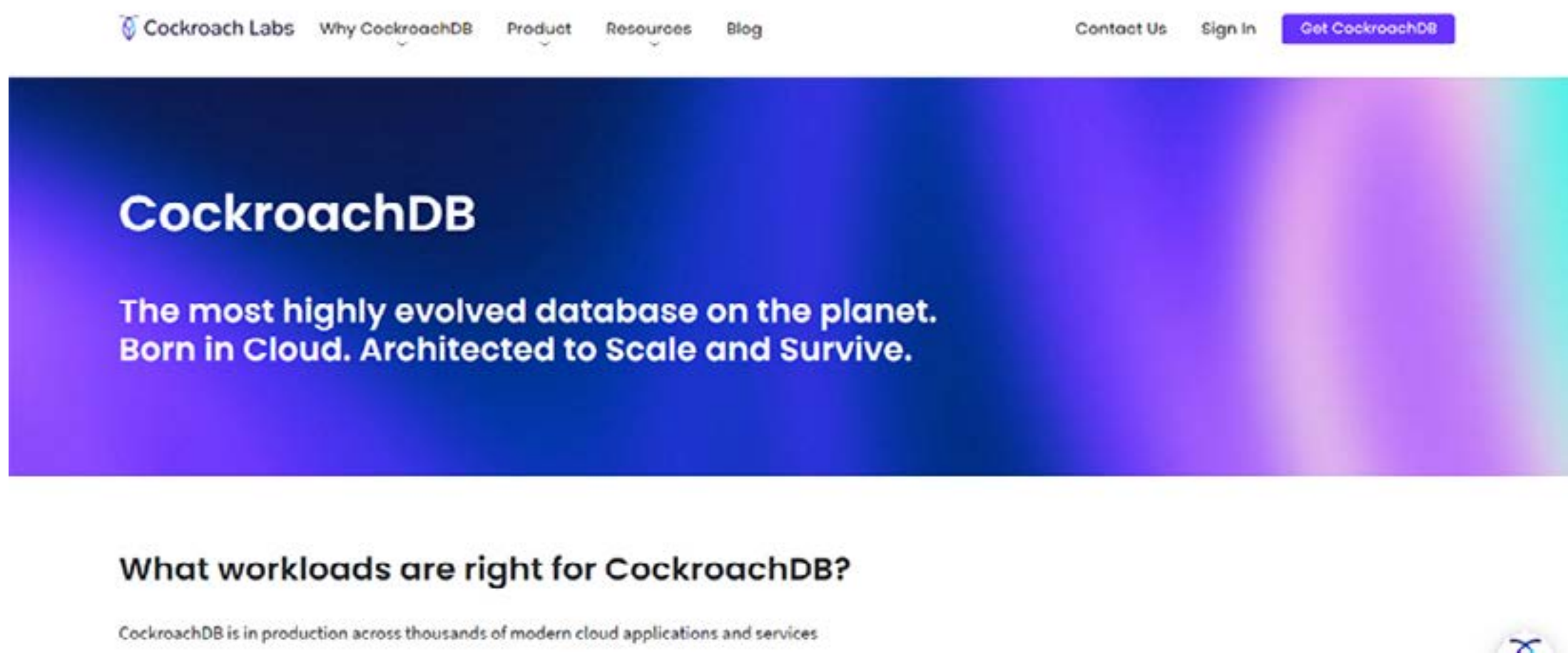


A característica mais interessante deste banco de dados é a oferta de 4 APIs diferentes de manipulação de dados: para as tabelas do Azure (chave-Valor), para Gremlin (grafos), para MongoDB (JSON) e Cassandra (colunar). Também possui integração com .NET, JavaScript, Python e C++. Uma gama de opções bastante interessante no desenvolvimento de aplicações.

E sobre o Cockroach DB, você já ouviu falar? Confira, a seguir, as suas características.

## 1.7.4 Cockroach DB

O CockroachDB é utilizado por grandes corporações como Nubank e Comcast e é um representante de mais uma evolução tecnológica em banco de dados.



Fonte: CockroachDB Products

Até este momento, tivemos a oportunidade de conhecer basicamente dois “mundos” em banco de dados: o Relacional e o NoSql. Aprendemos, também, que, em aplicações específicas, em favor de desempenho, alguns controles de integridade e de concorrência tiveram que ser flexibilizados.

Agora, surge um movimento tecnológico em prol de unir o desempenho e escalabilidade dos bancos NoSql sem ter que abrir mão das propriedades ACID e dos benefícios oferecidos pelos bancos de dados relacionais tradicionais. Esses novos modelos de bancos de dados têm sido referenciados como NewSql.

Em busca deste objetivo, a ideia é que, ao contrário dos tradicionais bancos relacionais que foram

concebidos como uma única solução para qualquer tipo de aplicação, os bancos NewSql sejam especialistas, de forma autônoma, em um propósito específico.

Para que um banco de dados seja considerado NewSql, são definidas cinco características, conforme apontam Stonebraker e Cattel (2011):

- **linguagem SQL como meio de interação entre o SGBD e a aplicação;**
- **suporte a transações ACID;**
- **controle de concorrência sem bloqueios em leituras e escritas concorrentes;**
- **arquitetura que forneça um maior desempenho por nó de processamento;**
- **arquitetura escalável, com memória distribuída e com capacidade de funcionar com um grande número de nós.**

Quer entender um pouco mais sobre os bancos de dados mais utilizados no momento?

Acesse o portal DB-Engines Ranking, mantido pela consultoria austríaca Solid IT.

O portal utiliza informações dos mecanismos de busca, Google Trends, fóruns de discussões, sites de ofertas de emprego, Twitter e perfis profissionais como o LinkedIn para classificar e medir tendências em bancos de dados. Nele, você encontrará um ranking aplicado a todos os bancos de dados; **um ranking aplicado aos bancos de dados** relacionais; e um ranking aplicado aos **bancos de dados NoSql** do tipo documento.

Para finalizar, há outros tipos de bancos que você precisa conhecer. Vamos lá?

## 1.8 Outros tipos de banco de dados

Você conheceu alguns tipos de bancos de dados nos tópicos anteriores. Mas há outros bancos de dados que atendem requisitos muito específicos, e que oferecem soluções que justificam termos um breve conhecimento sobre eles. Veja a seguir!

**Bancos de dados temporais:** armazenam e recuperam informações ligadas a sua evolução no tempo. O SGBD executa o gerenciamento temporal da informação através de seus estados presente, passado e futuro. São especialmente úteis, por exemplo, no armazenamento de histórico

de pacientes e controle acadêmico. Um exemplo deste tipo de SGBD é o banco de dados de código aberto InfluxDb.

**Bancos de dados geográficos:** são específicos para a manipulação de informações espaciais e geográficas. Devem lidar com cálculos de área, comprimentos e ângulos, projeção de dados e operações geométricas. Podem ser utilizados em sistemas de monitoramento de uso do solo, censos populacionais e estudos ambientais. Para essa finalidade, o banco de dados Postgresql possui a extensão Postgis e o Oracle a extensão Spatial.

**Bancos de dados multidimensionais:** aqui, o termo não se refere ao SGBD, mas sim a uma técnica de modelagem. O principal objetivo é produzir múltiplas visões (dimensões) a respeito de um fato. Essa técnica já foi mencionada no início do capítulo e é utilizada em aplicações de Business Intelligence.

Podemos tomar como exemplo um fato “cliente”. Esta modelagem, cria uma tabela central com as informações cadastrais, e tabelas secundárias contendo todos os fatos que ocorrem no relacionamento com o cliente. Esse esquema, com uma tabela principal (fato) e várias outras secundárias (dimensões) ligadas a ela, é conhecido como esquema estrela. Este é o esquema projetado em aplicações de Data Warehouse e Business Intelligence. O mercado oferece várias ferramentas que a partir desta armazenagem geram múltiplas visões destes relacionamentos com o objeto da tabela de fatos. No nosso exemplo, o cliente.

Pudemos, nesta unidade, adquirir conhecimentos sobre as opções tecnológicas e de negócio na área de bancos de dados. Continue a se aprofundar nessa temática!



# Síntese

Nesta unidade, vimos que a escolha pela melhor opção de armazenagem e banco de dados está intimamente relacionada às necessidades a serem atendidas, e que cada tecnologia tem um foco maior em requisitos específicos. Assim, confira a seguir os itens mais importantes.

- Percebemos que tecnologias e modelos de negócio podem ser combinados em favor de melhor eficiência e desempenho das aplicações.
- Bancos de dados relacionais são especialistas em aplicações intensivas em I/O e em segurança de transações concorrentes.
- Bancos de dados distribuídos apresentam uma boa relação custo-benefício na escalabilidade de bancos de dados, porém demandam atenção no controle de concorrência.
- Serviços de armazenamento de dados em nuvem podem ser uma ótima opção, uma vez que a complexidade de gerenciamento e implementação pode ficar sob a expertise do fornecedor.
- Bancos de dados NoSql atentem à necessidade de análise e manipulação de grandes volumes de informação em dados não estruturados ou em aplicações de baixa latência.
- Alguns dos bancos de dados despontam como muito promissores, como o Amazon DynamoDB, o Google Bigtable e BigQuery, o Microsoft CosmosDb e o CockroachDb.

# Fulture Insights

ELMASRI, R.; NAVATHE, S. B. Sistemas de Banco de Dados. 7. ed. Bonston: Pearson-Addison-Wesley, 2018.

STONEBRAKER, M., CATTELL, R. 10 Rules for Scalable Performance in 'Simple Operation' Datastores. Communications of the ACM, v. 54, n. 6(Jun), pp 72-80, 2011. Disponível em: [https://www.researchgate.net/publication/220423938\\_10\\_Rules\\_for\\_Scalable\\_Performance\\_in\\_'Simple\\_Operation'\\_Datastores](https://www.researchgate.net/publication/220423938_10_Rules_for_Scalable_Performance_in_'Simple_Operation'_Datastores). Acesso em: 09 mar. 2021.

FU  
LL  
TU  
RE

[www.fulture.com](http://www.fulture.com)