Decomposição do banco de dados em AWS

AWS Orientação prescritiva

AWS Orientação prescritiva: Decomposição do banco de dados em AWS

Table of Contents

Decomposição do banco de dados em AWS

Philippe Wanner e Saurabh Sharma, Amazon Web Services

Outubro de 2025 (histórico do documento)

A modernização do banco de dados, especialmente a decomposição de bancos de dados monolíticos, é um fluxo de trabalho essencial para organizações que desejam melhorar a agilidade, a escalabilidade e o desempenho em seus sistemas de gerenciamento de dados. À medida que as empresas crescem e suas necessidades de dados se tornam mais complexas, os bancos de dados monolíticos tradicionais geralmente têm dificuldade em acompanhar o ritmo. Isso leva a gargalos de desempenho, desafios de manutenção e dificuldade de adaptação às mudanças nos requisitos de negócios.

A seguir estão os desafios comuns com bancos de dados monolíticos:

- Desalinhamento do domínio comercial — Os bancos de dados monolíticos geralmente não conseguem alinhar a tecnologia com domínios comerciais distintos, o que pode limitar o crescimento organizacional.
- Restrições de escalabilidade — Os sistemas frequentemente atingem limites de escalabilidade, o que cria barreiras à expansão dos negócios.
- Rigidez arquitetônica — estruturas fortemente acopladas dificultam a atualização de componentes específicos sem afetar todo o sistema.
- Degradação do desempenho — O aumento das cargas de dados e o aumento da simultaneidade de usuários geralmente levam à deterioração do desempenho do sistema.

A seguir estão os benefícios da decomposição do banco de dados:

- Agilidade comercial aprimorada — a decomposição permite uma adaptação rápida às mudanças nas necessidades comerciais e oferece suporte ao dimensionamento independente.
- Desempenho otimizado — A decomposição ajuda você a criar soluções de banco de dados especializadas que são personalizadas para casos de uso específicos e escalam cada banco de dados de forma independente.
- Melhor gerenciamento de custos — A decomposição permite uma utilização mais eficiente dos recursos e reduz os custos operacionais.

- Opções flexíveis de licenciamento — A decomposição cria oportunidades de transição de licenças proprietárias caras para alternativas de código aberto.
- Viabilização da inovação — A decomposição facilita a adoção de bancos de dados específicos para cargas de trabalho específicas.

Público-alvo

Este guia ajuda arquitetos de banco de dados, arquitetos de soluções de nuvem, equipes de desenvolvimento de aplicativos e arquitetos corporativos. Ele foi projetado para ajudá-lo a decompor bancos de dados monolíticos em armazenamentos de dados alinhados a microsserviços, implementar arquiteturas de banco de dados orientadas por domínio, planejar estratégias de migração de banco de dados e escalar as operações de banco de dados para atender às crescentes demandas comerciais. Para entender os conceitos e recomendações deste guia, você deve estar familiarizado com os princípios do banco de dados relacional e NoSQL AWS , os serviços de banco de dados gerenciados e os padrões de arquitetura de microsserviços. Este guia tem como objetivo ajudar organizações que estão nos estágios iniciais de um projeto de decomposição de banco de dados.

Objetivos

Este guia pode ajudar sua organização a alcançar os seguintes objetivos:

- Colete os requisitos para decompor sua arquitetura de destino.
- Desenvolva uma metodologia sistemática para avaliar riscos e se comunicar.
- Crie um plano de decomposição.
- Defina métricas de sucesso, indicadores-chave de desempenho (KPIs), uma estratégia de mitigação e um plano de continuidade de negócios.
- Estabeleça uma melhor elasticidade da carga de trabalho que ajude você a acompanhar a demanda dos negócios.
- Saiba como adotar bancos de dados especializados para casos de uso específicos, o que possibilita a inovação.
- Fortaleça a segurança e a governança de dados da sua organização.
- Reduza os custos por meio do seguinte:
  - Taxas de licenciamento reduzidas

- Redução da dependência de fornecedores
- Acesso aprimorado a inovações e suporte mais amplos da comunidade
- Capacidade de escolher diferentes tecnologias de banco de dados para diferentes componentes
- Migração gradual, que reduz o risco e distribui os custos ao longo do tempo
- Melhor utilização de recursos

Desafios comuns e responsabilidades de gerenciamento pela decomposição do banco de dados

A decomposição do banco de dados é um processo complexo que requer planejamento, execução e gerenciamento cuidadosos. À medida que as organizações buscam modernizar sua infraestrutura de dados, elas geralmente enfrentam uma infinidade de desafios que podem afetar o sucesso de seus projetos. Esta seção descreve os obstáculos comuns e apresenta uma abordagem estruturada para superá-los.

Desafios comuns

Um projeto de decomposição de banco de dados enfrenta vários desafios nas dimensões técnica, pessoal e comercial. Em termos técnicos, garantir a consistência dos dados em sistemas distribuídos representa um obstáculo significativo. Também pode ter impactos potenciais no desempenho e na estabilidade durante o período de transição, e você deve se integrar perfeitamente aos sistemas existentes. Os desafios relacionados às pessoas incluem a curva de aprendizado associada ao novo sistema, a potencial resistência dos funcionários à mudança e a disponibilidade dos recursos necessários. Do ponto de vista comercial, o projeto deve lidar com os riscos de excesso de cronograma, restrições orçamentárias e o potencial de interrupção dos negócios durante o processo de migração.

Definindo funções e responsabilidades

Dados esses desafios complexos que abrangem as dimensões técnica, pessoal e comercial, estabelecer funções e responsabilidades claras se torna fundamental para o sucesso do projeto. Uma matriz responsável, responsável, consultada e informada (RACI) fornece a estrutura necessária para enfrentar esses desafios. Ele define explicitamente quem toma decisões, quem executa o trabalho, quem fornece informações e quem precisa se manter informado em cada estágio da decomposição. Essa clareza ajuda a evitar atrasos causados por tomadas de decisão ambíguas, incentiva o engajamento adequado das partes interessadas e cria responsabilidade pelos principais resultados. Sem essa estrutura, as equipes podem ter dificuldades com responsabilidades sobrepostas, comunicações perdidas e caminhos de escalonamento pouco claros — problemas que podem agravar as complexidades técnicas existentes e os desafios de gerenciamento de mudanças, ao mesmo tempo em que aumentam o risco de excesso de cronograma e orçamento.

O exemplo de matriz RACI a seguir é um ponto de partida que pode ajudá-lo a esclarecer possíveis funções e responsabilidades em sua organização.

Tarefa ou atividade	Gerente de projeto	Arquiteto	Desenvolvedor	Parte interessada
Identifique resultados e desafios de negócios	A/R	R	C	–
Defina o escopo e identifique os requisitos	A	R	C	C/I
Identifique as métricas de sucesso do projeto	A	R	C	eu
Crie e execute o plano de comunicação	A/R	C	C	eu
Defina a arquitetura de destino	eu	A/R	C	–
Controle o acesso ao banco de dados	eu	A/R	R	–
Crie e execute o plano de continuidade de negócios	A/R	C	eu	–
Analise a coesão e o acoplamento	eu	A/R	R	eu

Mova a lógica de negócios (como procedimentos armazenados) do banco de dados para a camada do aplicativo	eu	A	R	–
Separe as relações de tabela, conhecidas como junções	eu	A	R	–

Definindo o escopo e os requisitos para a decomposição do banco de dados

Ao definir o escopo e identificar os requisitos para seu projeto de decomposição do banco de dados, você deve retroceder às necessidades da sua organização. Isso requer uma abordagem sistemática que equilibre a viabilidade técnica com o valor comercial. Essa etapa inicial define a base de todo o processo e ajuda a garantir que os objetivos do projeto estejam alinhados às metas e capacidades da organização.

Esta seção contém os seguintes tópicos:

Estabelecendo uma estrutura de análise central

A definição do escopo começa com um fluxo de trabalho sistemático que orienta a análise em quatro fases interconectadas. Essa abordagem abrangente garante que os esforços de decomposição do banco de dados sejam fundamentados em uma compreensão completa dos sistemas existentes e dos requisitos operacionais. A seguir estão as fases da estrutura de análise principal:

1. Análise de atores — Identifique minuciosamente todos os sistemas e aplicativos que interagem com o banco de dados. Isso envolve mapear tanto os produtores que realizam operações de gravação quanto os consumidores que lidam com as operações de leitura, documentando seus padrões de acesso, frequências e horários de pico de uso. Essa visão centrada no cliente ajuda você a entender o impacto de qualquer mudança e a identificar caminhos críticos que exigem atenção especial durante a decomposição.
2. Análise de atividades — Mergulhe profundamente nas operações específicas que cada ator realiza. Você cria matrizes detalhadas de criação, leitura, atualização e exclusão (CRUD) para cada sistema e identifica quais tabelas elas acessam e como. Essa análise ajuda a descobrir

limites naturais para a decomposição e destaca áreas nas quais você pode simplificar a arquitetura atual.

3. Mapeamento de dependências — documente dependências diretas e indiretas entre sistemas, criando visualizações claras dos fluxos e relacionamentos de dados. Isso ajuda a identificar possíveis pontos de ruptura e áreas em que é necessário um planejamento cuidadoso para ganhar confiança. A análise considera dependências técnicas, como tabelas compartilhadas e chaves estrangeiras, e dependências de processos de negócios, como sequências de fluxo de trabalho e requisitos de relatórios.
4. Requisitos de consistência — examine as necessidades de consistência de cada operação com altos padrões. Determine quais operações exigem consistência imediata, como transações financeiras. Outras operações podem operar com consistência eventual, como atualizações de análises. Essa análise influencia diretamente a escolha dos padrões de decomposição e as decisões arquitetônicas em todo o projeto.

Definindo limites do sistema para decomposição do banco de dados

Os limites do sistema são perímetros lógicos que definem onde um sistema termina e outro começa, abrangendo propriedade de dados, padrões de acesso e pontos de integração. Ao definir os limites do sistema, faça escolhas ponderadas, mas decisivas, que equilibrem o planejamento abrangente com as necessidades práticas de implementação. Considere o banco de dados como uma unidade lógica que pode abranger vários bancos de dados físicos ou esquemas. Essa definição de limite cumpre os seguintes objetivos críticos:

- Identifica todos os atores externos e seus padrões de interação
- Mapeia de forma abrangente as dependências de entrada e saída
- Documenta restrições técnicas e operacionais
- Delineia claramente o escopo do esforço de decomposição

Considerando os ciclos de lançamento

Compreender os ciclos de lançamento é crucial para planejar a decomposição do banco de dados. Revise os tempos de renovação do sistema de destino e de qualquer sistema dependente. Identifique oportunidades para mudanças coordenadas. Considere qualquer descomissionamento planejado de sistemas conectados, pois isso pode influenciar sua estratégia de decomposição.

Considere as janelas de mudança e as restrições de implantação existentes para minimizar a interrupção dos negócios. Certifique-se de que seu plano de implementação esteja alinhado com os cronogramas de lançamento em todos os sistemas conectados.

Avaliação das restrições técnicas para a decomposição do banco de dados

Antes de prosseguir com a decomposição do banco de dados, avalie as principais limitações técnicas que moldarão sua abordagem de modernização. Examine os recursos de sua pilha de tecnologia atual, incluindo versões de banco de dados, estruturas, requisitos de desempenho e contratos de nível de serviço. Considere os mandatos de segurança e conformidade, especialmente para setores regulamentados. Analise os volumes de dados atuais, as projeções de crescimento e as ferramentas de migração disponíveis para embasar suas decisões de escalabilidade. Por fim, confirme seus direitos de acesso ao código-fonte e às modificações do sistema, pois eles determinarão as estratégias de decomposição viáveis.

Entendendo o contexto organizacional

A decomposição bem-sucedida do banco de dados exige que você entenda o cenário organizacional mais amplo no qual o sistema opera. Mapeie dependências entre departamentos e estabeleça canais de comunicação claros entre as equipes. Avalie as capacidades técnicas da sua equipe e identifique quaisquer necessidades de treinamento ou lacunas de habilidades que você precise resolver. Considere as implicações do gerenciamento de mudanças, inclusive como gerenciar as transições e manter a continuidade dos negócios. Avalie os recursos disponíveis e quaisquer restrições, como limitações orçamentárias ou de pessoal. Por fim, alinhe sua estratégia de decomposição às expectativas e prioridades das partes interessadas para promover o apoio contínuo durante todo o projeto.

Avaliação do risco de decomposição do banco de dados

Uma avaliação de risco abrangente é essencial para o sucesso da decomposição do banco de dados. Avalie cuidadosamente os riscos, como integridade dos dados durante a migração, possível degradação do desempenho do sistema, possíveis falhas de integração e vulnerabilidades de segurança. Esses desafios técnicos devem ser equilibrados com os riscos comerciais, incluindo possíveis interrupções operacionais, limitações de recursos, atrasos no cronograma e restrições orçamentárias. Para cada risco identificado, desenvolva estratégias de mitigação e planos de

contingência específicos para manter a dinâmica do projeto e, ao mesmo tempo, proteger as operações comerciais.

Crie uma matriz de risco que avalie o impacto e a probabilidade de possíveis problemas. Trabalhe com equipes técnicas e partes interessadas do negócio para identificar riscos, definir limites claros para intervenção e desenvolver estratégias específicas de mitigação. Por exemplo, classifique o risco de perda de dados como alto impacto e baixa probabilidade, e isso requer estratégias de backup robustas. Uma pequena degradação do desempenho pode ter impacto médio e alta probabilidade, além de exigir monitoramento proativo.

Estabeleça ciclos regulares de análise de risco para reavaliar as prioridades e ajustar os planos de mitigação à medida que o projeto evolui. Essa abordagem sistemática garante que os recursos se concentrem nos riscos mais críticos, mantendo caminhos claros de escalonamento para problemas emergentes.

Definindo critérios de sucesso para a decomposição do banco de dados

Os critérios de sucesso para a decomposição do banco de dados devem ser claramente definidos e mensuráveis em várias dimensões. Do ponto de vista comercial, estabeleça metas específicas para redução de custos, melhoria time-to-market, disponibilidade do sistema e satisfação do cliente. O sucesso técnico deve ser medido por meio de melhorias quantificáveis no desempenho do sistema, na eficiência da implantação, na consistência dos dados e na confiabilidade geral. Para o processo de migração, defina requisitos rigorosos para zero perda de dados, limites aceitáveis de interrupção dos negócios, conformidade orçamentária e cumprimento do cronograma.

Documente esses critérios minuciosamente, mantendo métricas básicas e metas, metodologias de medição claras e cronogramas de revisão regulares. Atribua proprietários claros para cada métrica de sucesso e mapeie dependências entre métricas diferentes. Essa abordagem abrangente para medir o sucesso alinha as conquistas técnicas aos resultados comerciais, ao mesmo tempo em que mantém a responsabilidade durante toda a jornada de decomposição.

Controle do acesso ao banco de dados durante a decomposição

Muitas organizações enfrentam um cenário comum: um banco de dados central que cresceu organicamente ao longo de muitos anos e é acessado diretamente por vários serviços e equipes. Isso cria vários problemas críticos:

- Crescimento descontrolado — À medida que as equipes adicionam continuamente novos recursos e modificam esquemas, o banco de dados se torna cada vez mais complexo e difícil de gerenciar.
- Preocupações com o desempenho — Mesmo com melhorias de hardware, a carga crescente acaba ameaçando exceder os recursos do banco de dados. Impossibilidade de ajustar as consultas devido à complexidade do esquema ou à falta de habilidades. Não é possível prever ou explicar o desempenho do sistema.
- Paralisia de decomposição — Torna-se quase impossível dividir ou refatorar o banco de dados enquanto ele está sendo ativamente modificado por várias equipes.

> **Note**
>
> Os sistemas de banco de dados monolíticos geralmente reutilizam as mesmas credenciais para aplicativos, serviços ou para administração. Isso leva a uma baixa rastreabilidade do banco de dados. Definir funções dedicadas e adotar o princípio do menor privilégio pode ajudá-lo a aumentar a segurança e a disponibilidade.

Ao lidar com um banco de dados monolítico que se tornou complicado, um dos padrões mais eficazes para controlar o acesso é chamado de serviço de encapsulamento de banco de dados. Ele fornece uma primeira etapa estratégica no gerenciamento de sistemas complexos de banco de dados. Ele estabelece acesso controlado ao banco de dados e permite a modernização gradual, ao mesmo tempo em que reduz os riscos. Essa abordagem cria uma base para melhorias incrementais, fornecendo visibilidade clara dos padrões e dependências de uso de dados. É uma arquitetura de transição que serve como um passo em direção à decomposição completa do banco de dados. O serviço de embalagem fornece a estabilidade e o controle necessários para fazer essa viagem com sucesso.

Esta seção contém os seguintes tópicos:

Controlando o acesso com o padrão de serviço do invólucro do banco de dados

Um serviço de wrapper é uma camada de serviço que atua como uma fachada para o banco de dados. Essa abordagem é particularmente valiosa quando você precisa manter a funcionalidade existente enquanto se prepara para a futura decomposição. Esse padrão segue um princípio simples: quando algo está muito confuso, comece por conter a bagunça. O serviço de wrapper se torna a única forma autorizada de acessar o banco de dados, fornecendo uma interface controlada e ocultando a complexidade subjacente.

Use esse padrão quando a decomposição imediata do banco de dados não for viável devido a esquemas complexos ou quando vários serviços exigirem acesso contínuo aos dados. É particularmente valioso durante os períodos de transição porque fornece tempo para uma refatoração cuidadosa, mantendo a estabilidade do sistema. O padrão funciona bem ao consolidar a propriedade de dados para equipes específicas ou quando novos aplicativos precisam de visualizações agregadas em várias tabelas.

Por exemplo, aplique esse padrão quando:

- A complexidade do esquema impede a separação imediata
- Várias equipes precisam de acesso contínuo aos dados
- A modernização gradual é preferida
- A reestruturação da equipe exige uma propriedade clara dos dados
- Novos aplicativos precisam de visualizações de dados consolidadas

Benefícios e limitações do padrão de serviço do wrapper de banco de dados

A seguir estão os benefícios do padrão de invólucro do banco de dados:

- Crescimento controlado — O serviço de wrapper evita novas adições não controladas ao esquema do banco de dados.

- Limites claros — O processo de implementação ajuda você a estabelecer limites claros de propriedade e responsabilidade.
- Liberdade de refatoração — Um serviço de embalagem permite que você faça alterações internas sem afetar os consumidores.
- Observabilidade aprimorada — Um serviço de empacotamento é um ponto único para monitoramento e registro.
- Teste simplificado — Um serviço de wrapper facilita o consumo de serviços para criar versões simplificadas e simuladas para testes.

A seguir estão as limitações do padrão de invólucro do banco de dados.

- Acoplamento de tecnologia — Um serviço de empacotamento funciona melhor quando usa a mesma pilha de tecnologia dos serviços consumidores.
- Sobrecarga inicial — O serviço de empacotamento requer infraestrutura adicional que pode afetar o desempenho.
- Esforço de migração — Para implementar o serviço de embalagem, você deve coordenar as equipes para sair do acesso direto.
- Desempenho — se o serviço de embalagem apresentar alto tráfego, uso intenso ou acesso frequente, os serviços de consumo podem apresentar baixo desempenho. Além do banco de dados, o serviço de wrapper deve lidar com paginação, cursores e conexões de banco de dados. Dependendo do seu caso de uso, ele pode não ser bem dimensionado e pode ser inadequado para cargas de trabalho de extração, transformação e carregamento (ETL).

Implementando o padrão de serviço do wrapper de banco de dados

Há duas fases para implementar o padrão de serviço do wrapper de banco de dados. Primeiro, você cria o serviço de encapsulamento de banco de dados. Em seguida, você direciona todo o acesso por meio dele e documenta os padrões de acesso.

Fase 1: Criação do serviço de encapsulamento de banco de dados

Crie uma camada de serviço leve que atue como guardiã do seu banco de dados. Inicialmente, ele deve espelhar todas as funcionalidades existentes. Esse serviço de wrapper se torna o ponto de acesso obrigatório para todas as operações do banco de dados, o que converte dependências diretas do banco de dados em dependências de nível de serviço. Implemente registro e monitoramento detalhados nessa camada para rastrear padrões de uso, métricas de desempenho

e frequências de acesso. Mantenha seus procedimentos armazenados existentes, mas certifique-se de que eles sejam acessados somente por meio dessa nova interface de serviço.

Fase 2: Implementação do controle de acesso

Redirecione sistematicamente todo o acesso ao banco de dados por meio do serviço wrapper e, em seguida, revogue as permissões diretas do banco de dados de sistemas externos que acessam o banco de dados diretamente. Documente cada padrão de acesso e dependência à medida que os serviços são migrados. Esse acesso controlado permite a refatoração interna dos componentes do banco de dados sem interromper os consumidores externos. Por exemplo, comece com operações de baixo risco e somente para leitura, em vez de fluxos de trabalho transacionais complexos.

Fase 3: Monitorar o desempenho do banco de dados

Use o serviço wrapper como um ponto de monitoramento centralizado para o desempenho do banco de dados. Acompanhe as principais métricas, incluindo tempos de resposta de consultas, padrões de uso, taxas de erro e utilização de recursos. Configure alertas para limites de desempenho e padrões incomuns. Por exemplo, monitore consultas de execução lenta, utilização do pool de conexões e taxa de transferência de transações para identificar proativamente possíveis problemas.

Use essa visualização consolidada para otimizar o desempenho do banco de dados por meio de ajustes de consultas, ajustes de alocação de recursos e análise de padrões de uso. A natureza centralizada do serviço de embalagem facilita a implementação de melhorias e a validação de seu impacto em todos os consumidores, mantendo padrões de desempenho consistentes.

Melhores práticas para implementar um serviço de encapsulamento de banco de dados

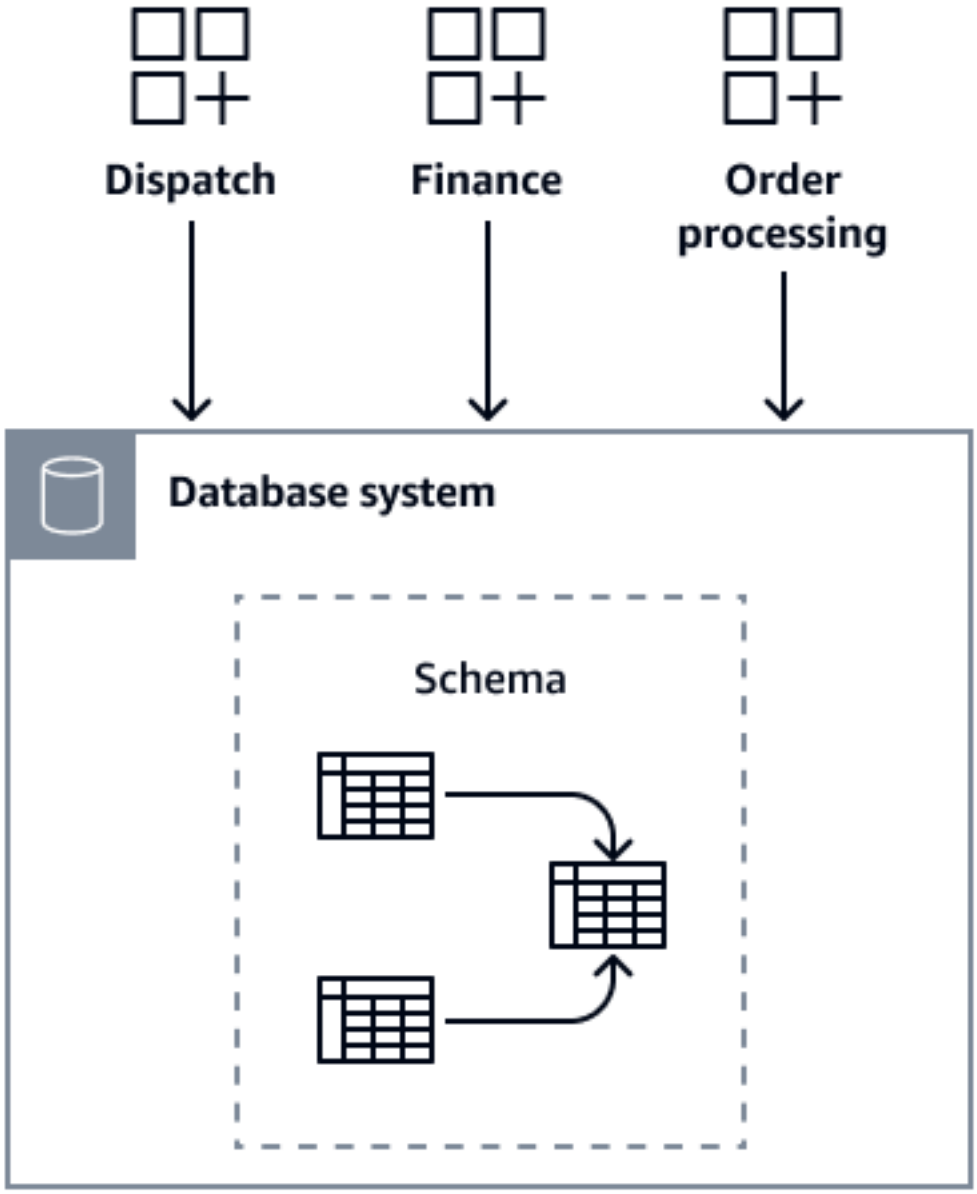
As práticas recomendadas a seguir podem ajudá-lo a implementar um serviço de encapsulamento de banco de dados:

- Comece pequeno — comece com um invólucro mínimo que simplesmente serve como proxy para a funcionalidade existente
- Mantenha a estabilidade — mantenha a interface de serviço estável enquanto faz melhorias internas
- Monitore o uso — implemente um monitoramento abrangente para entender os padrões de acesso
- Propriedade clara — designe uma equipe dedicada para manter o invólucro e o esquema subjacente

- Incentive o armazenamento local — Motive as equipes a armazenar seus dados em seus próprios bancos de dados

Exemplo baseado em cenários

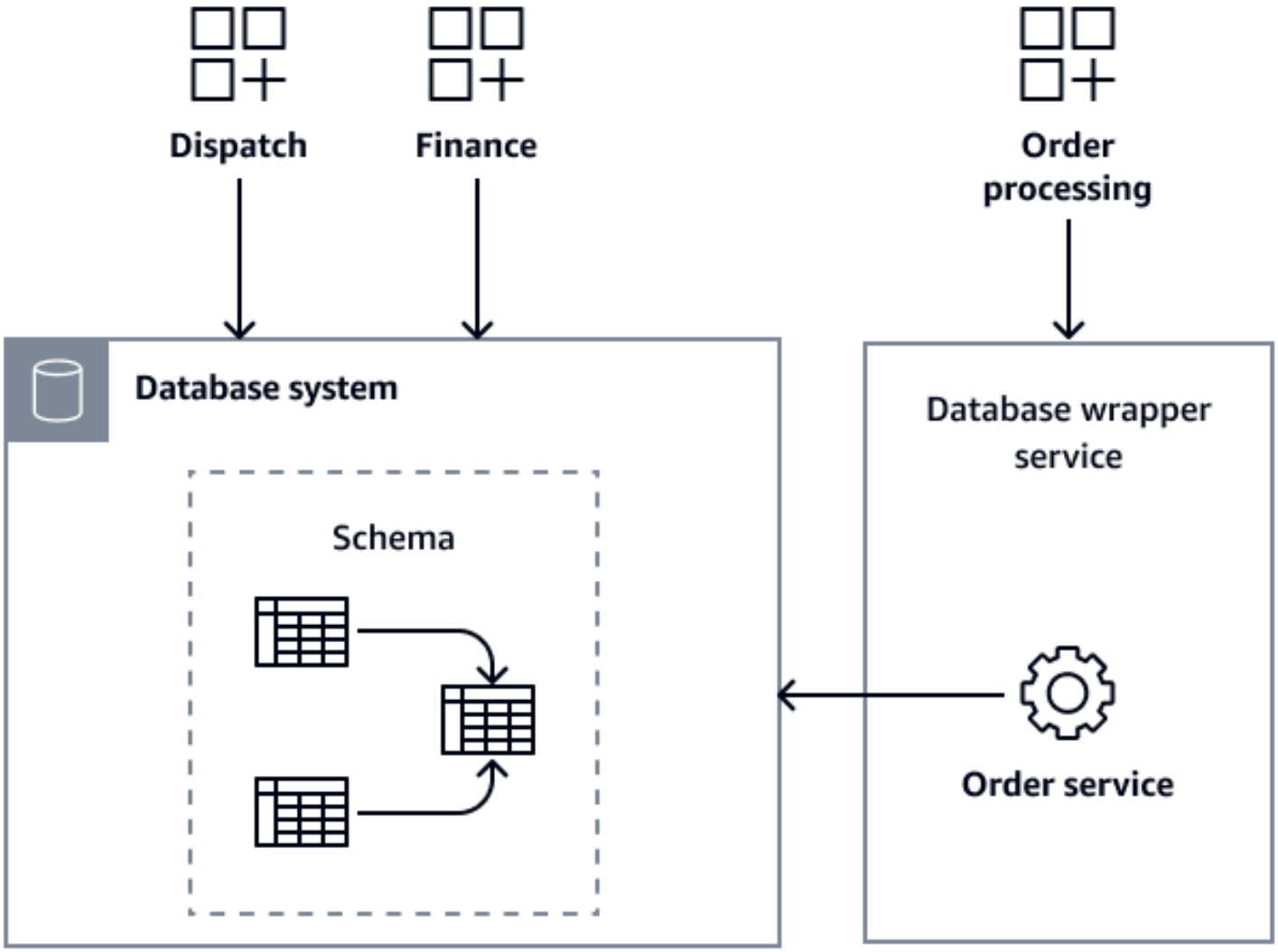
Esta seção descreve um exemplo de como uma empresa fictícia, chamada AnyCompany Books, poderia usar o padrão de invólucro de banco de dados para controlar o acesso ao seu sistema de banco de dados monolítico. Na AnyCompany Books, existem três serviços essenciais: despacho, finanças e processamento de pedidos. Esses serviços compartilham o acesso a um banco de dados central. Cada serviço é mantido por uma equipe diferente. Com o tempo, eles modificam de forma independente o esquema do banco de dados para atender às suas necessidades específicas. Isso levou a uma rede emaranhada de dependências e a uma estrutura de banco de dados cada vez mais complexa.



O arquiteto corporativo ou de aplicativos da empresa reconhece a necessidade de decompor esse banco de dados monolítico. O objetivo deles é fornecer a cada serviço seu próprio banco de dados dedicado para melhorar a capacidade de manutenção e reduzir as dependências entre equipes. No entanto, eles enfrentam um desafio significativo: é quase impossível decompor o banco de dados enquanto as três equipes continuam a modificá-lo ativamente para seus projetos em andamento. As constantes mudanças de esquema e a falta de coordenação entre as equipes tornam extremamente arriscado tentar qualquer reestruturação significativa.

O arquiteto usa o padrão de serviço do wrapper do banco de dados para começar a controlar o acesso ao banco de dados monolítico. Primeiro, eles configuraram o serviço de encapsulamento

de banco de dados para um módulo específico, chamado serviço de pedidos. Em seguida, eles redirecionam o serviço de processamento de pedidos para acessar o serviço de embalagem em vez de acessar diretamente o banco de dados. A imagem a seguir mostra a infraestrutura modificada.

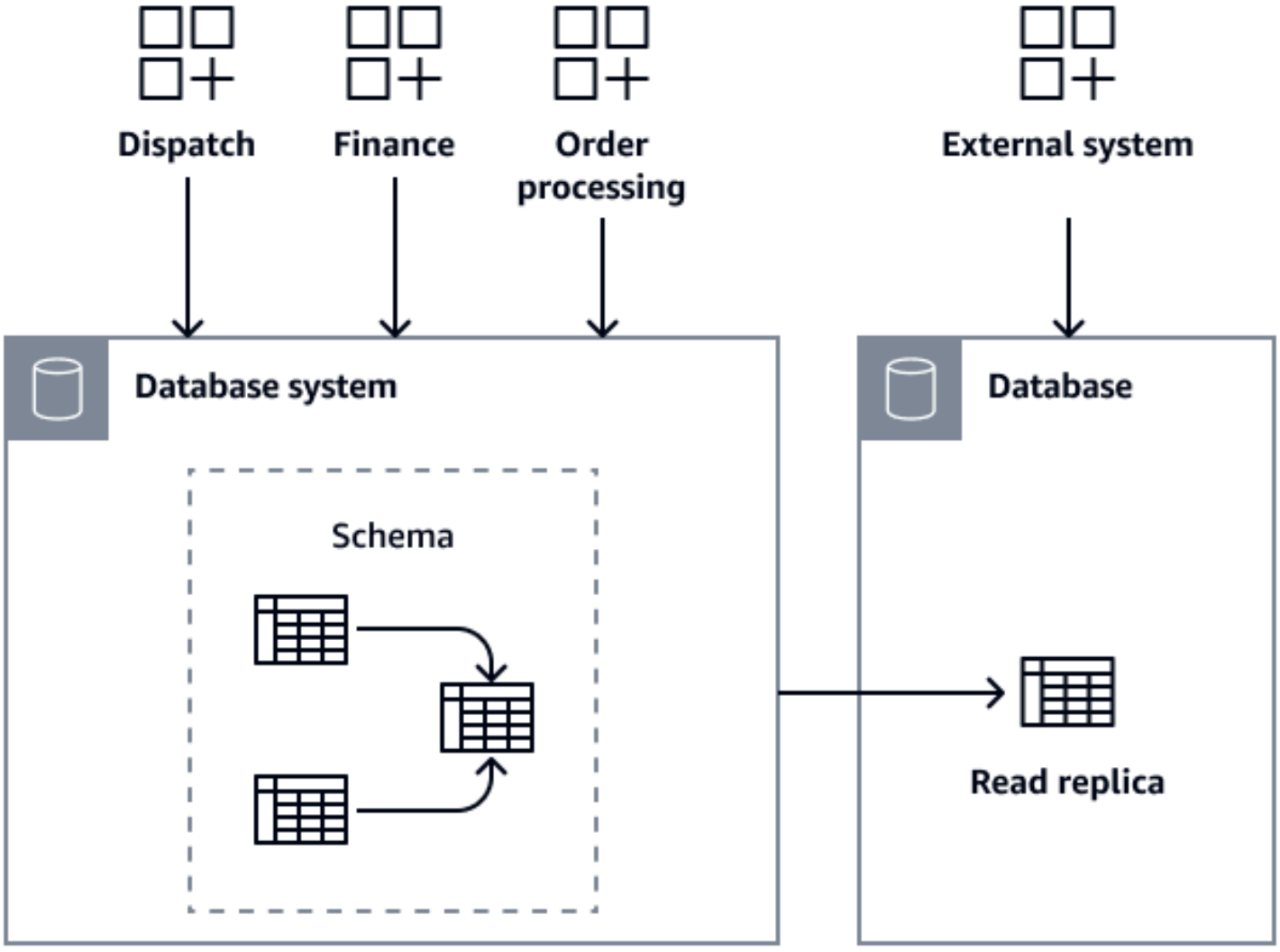


Controlando o acesso com o padrão CQRS

Outro padrão que você pode usar para isolar sistemas externos que se conectam a esse banco de dados central é a segregação de responsabilidade por consulta de comando (CQRS). Se alguns dos sistemas externos estiverem se conectando ao seu banco de dados central principalmente para leituras, como análises, relatórios ou outras operações de leitura intensa, você poderá criar armazenamentos de dados separados com otimização de leitura.

Esse padrão isola efetivamente esses sistemas externos dos impactos da decomposição do banco de dados e das alterações no esquema. Ao manter réplicas de leitura dedicadas ou

armazenamentos de dados específicos para padrões de consulta específicos, as equipes podem continuar suas operações sem serem afetadas pelas mudanças na estrutura principal do banco de dados. Por exemplo, enquanto você decompõe seu banco de dados monolítico, os sistemas de relatórios podem continuar trabalhando com suas visualizações de dados existentes, e as cargas de trabalho analíticas podem manter seus padrões de consulta atuais por meio de repositórios analíticos dedicados. Essa abordagem fornece isolamento técnico e permite autonomia organizacional porque equipes diferentes podem desenvolver seus sistemas de forma independente, sem se acoplar estreitamente à jornada de transformação do banco de dados principal.



Para obter mais informações sobre esse padrão e um exemplo de seu uso para desacoplar relações de tabela, consulte Padrão CQRS mais adiante neste guia.

Analisando a coesão e o acoplamento para decomposição do banco de dados

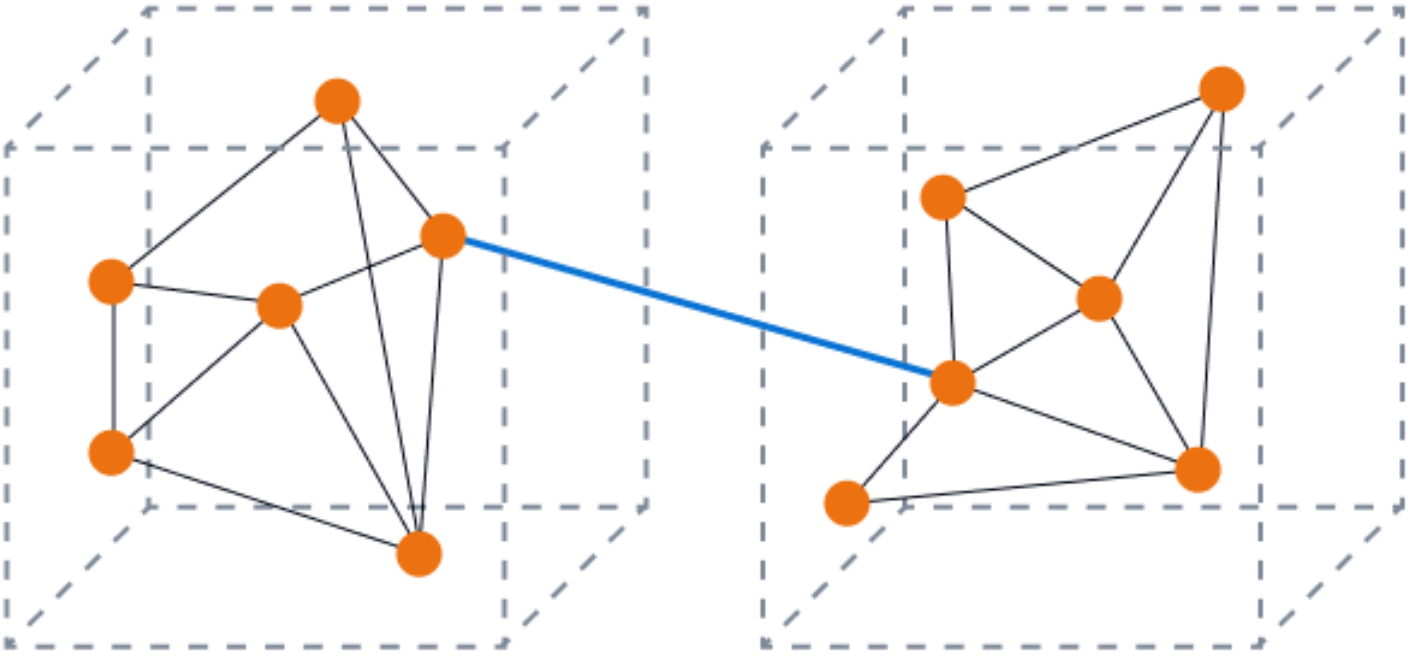
Esta seção ajuda você a analisar os padrões de acoplamento e coesão em seu banco de dados monolítico para orientar sua decomposição. Entender como os componentes do banco de dados interagem e dependem uns dos outros é crucial para identificar pontos de interrupção naturais, avaliar a complexidade e planejar uma abordagem de migração em fases. Essa análise revela dependências ocultas, destaca áreas que são adequadas para separação imediata e ajuda a priorizar os esforços de decomposição e, ao mesmo tempo, minimizar os riscos de transformação. Ao examinar o acoplamento e a coesão, você pode tomar decisões informadas sobre a sequência de separação dos componentes para manter a estabilidade do sistema durante todo o processo de transformação.

Esta seção contém os seguintes tópicos:

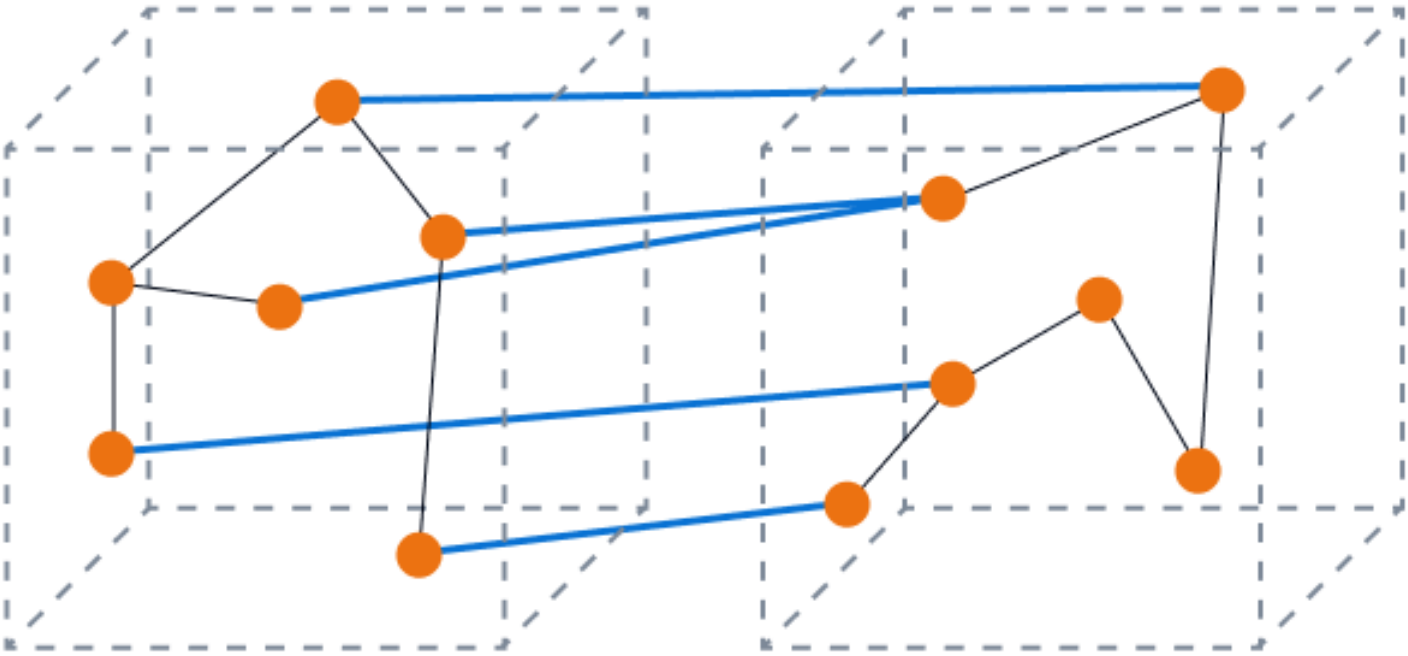
Sobre coesão e acoplamento

O acoplamento mede o grau de interdependência entre os componentes do banco de dados. Em um sistema bem projetado, você deseja obter um acoplamento frouxo, em que as alterações em um componente tenham um impacto mínimo sobre os outros. A coesão mede o quão bem os elementos dentro de um componente de banco de dados trabalham juntos para servir a um propósito único e bem definido. A alta coesão indica que os elementos de um componente estão fortemente relacionados e focados em uma função específica. Ao decompor um banco de dados monolítico, você deve analisar a coesão dentro dos componentes individuais e o acoplamento entre eles. Essa análise ajuda você a tomar decisões informadas sobre como detalhar o banco de dados e, ao mesmo tempo, manter a integridade e o desempenho do sistema.

A imagem a seguir mostra um acoplamento solto com alta coesão. Os componentes do banco de dados trabalham juntos para executar uma função específica e você minimiza o impacto da alteração em um único componente. Esse é o estado ideal.



A imagem a seguir mostra alto acoplamento com baixa coesão. Os componentes do banco de dados estão desconectados e é altamente provável que as alterações afetem outros componentes.



Padrões de acoplamento comuns em bancos de dados monolíticos

Há vários padrões de acoplamento que são comumente encontrados ao decompor um banco de dados monolítico em bancos de dados específicos de microsserviços. Compreender esses padrões

é crucial para iniciativas bem-sucedidas de modernização do banco de dados. Esta seção descreve cada padrão, seus desafios e as melhores práticas para reduzir o acoplamento.

Padrão de acoplamento de implementação

Definição: Os componentes estão fortemente interconectados no nível do código e do esquema. Por exemplo, modificar a estrutura de uma `customer` tabela `orderinventory`, impactos e `billing` serviços.

Impacto da modernização: cada microsserviço exige seu próprio esquema de banco de dados dedicado e camada de acesso a dados.

Desafios:

- Alterações nas tabelas compartilhadas afetam vários serviços
- Alto risco de efeitos colaterais não intencionais
- Aumento da complexidade dos testes
- Difícil modificar componentes individuais

Melhores práticas para reduzir o acoplamento:

- Defina interfaces claras entre os componentes
- Use camadas de abstração para ocultar detalhes de implementação
- Implemente esquemas específicos de domínio

Padrão de acoplamento temporal

Definição: As operações devem ser executadas em uma sequência específica. Por exemplo, o processamento de pedidos não pode continuar até que as atualizações de estoque sejam concluídas.

Impacto da modernização: cada microsserviço precisa de controle autônomo de dados.

Desafios:

- Quebrando dependências síncronas entre serviços
- Gargalos de desempenho
- Difícil de otimizar

- Processamento paralelo limitado

Melhores práticas para reduzir o acoplamento:

- Implemente o processamento assíncrono sempre que possível
- Use arquiteturas orientadas por eventos
- Projete para uma eventual consistência quando apropriado

Padrão de acoplamento de implantação

Definição: Os componentes do sistema devem ser implantados como uma única unidade. Por exemplo, uma pequena alteração na lógica de processamento de pagamentos exige a reimplantação de todo o banco de dados.

Impacto da modernização: implantações independentes de banco de dados por serviço

Desafios:

- Implantações de alto risco
- Frequência de implantação limitada
- Procedimentos complexos de reversão

Melhores práticas para reduzir o acoplamento:

- Divida em componentes que podem ser implantados de forma independente
- Implemente estratégias de fragmentação de banco de dados
- Use padrões de implantação azul-esverdeados

Padrão de acoplamento de domínio

Definição: Os domínios de negócios compartilham estruturas e lógica de banco de dados. Por exemplo, os `inventory` domínios `customerorder`, e compartilham tabelas e procedimentos armazenados.

Impacto da modernização: isolamento de dados específico do domínio

Desafios:

- Limites de domínio complexos
- Difícil escalar domínios individuais
- Regras de negócios confusas

Melhores práticas para reduzir o acoplamento:

- Identifique limites claros de domínio
- Dados separados por contexto de domínio
- Implemente serviços específicos de domínio

Padrões de coesão comuns em bancos de dados monolíticos

Há vários padrões de coesão que são comumente encontrados ao avaliar componentes do banco de dados para decomposição. Compreender esses padrões é crucial para identificar componentes de banco de dados bem estruturados. Esta seção descreve cada padrão, suas características e as melhores práticas para fortalecer a coesão.

Padrão de coesão funcional

Definição: Todos os elementos apoiam e contribuem diretamente para a execução de uma função única e bem definida. Por exemplo, todos os procedimentos e tabelas armazenados em um módulo de processamento de pagamentos tratam somente de operações relacionadas a pagamentos.

Impacto da modernização: padrão ideal para design de banco de dados de microsserviços

Desafios:

- Identificação de limites funcionais claros
- Separação de componentes de uso misto
- Manter uma responsabilidade única

Melhores práticas para fortalecer a coesão:

- Agrupe funções relacionadas
- Remover funcionalidades não relacionadas
- Defina limites claros dos componentes

Padrão de coesão sequencial

Definição: A saída de um elemento se torna entrada para outro. Por exemplo, os resultados da validação de um pedido são inseridos no processamento do pedido.

Impacto da modernização: requer análise cuidadosa do fluxo de trabalho e mapeamento do fluxo de dados

Desafios:

- Gerenciando dependências entre etapas
- Lidando com cenários de falha
- Mantendo a ordem do processo

Melhores práticas para fortalecer a coesão:

- Documente fluxos de dados claros
- Implemente o tratamento adequado de erros
- Crie interfaces claras entre as etapas

Padrão de coesão comunicacional

Definição: Os elementos operam com os mesmos dados. Por exemplo, todas as funções de gerenciamento do perfil do cliente funcionam com dados do cliente.

Impacto da modernização: ajuda a identificar limites de dados para separação de serviços a fim de diminuir o acoplamento entre os módulos

Desafios:

- Determinando a propriedade dos dados
- Gerenciando o acesso a dados compartilhados
- Mantendo a consistência dos dados

Melhores práticas para fortalecer a coesão:

- Defina uma propriedade clara dos dados
- Implemente padrões adequados de acesso aos dados

- Projete um particionamento de dados eficaz

Padrão de coesão processual

Definição: Os elementos são agrupados porque devem ser executados em uma ordem específica, mas podem não estar relacionados funcionalmente. Por exemplo, no processamento de pedidos, um procedimento armazenado que manipula a validação do pedido e a notificação do usuário é agrupado simplesmente porque eles acontecem em sequência, mesmo que tenham finalidades diferentes e possam ser tratados por serviços separados.

Impacto da modernização: exige uma separação cuidadosa dos procedimentos, mantendo o fluxo do processo

Desafios:

- Manter o fluxo correto do processo após a decomposição
- Identificação de limites funcionais reais em comparação com dependências processuais

Melhores práticas para fortalecer a coesão:

- Procedimentos separados com base em sua finalidade funcional e não na ordem de execução
- Use padrões de orquestração para gerenciar o fluxo do processo
- Implemente sistemas de gerenciamento de fluxo de trabalho para sequências complexas
- Projete arquiteturas orientadas a eventos para lidar com as etapas do processo de forma independente

Padrão de coesão temporal

Definição: Os elementos são relacionados por requisitos de tempo. Por exemplo, quando um pedido é feito, várias operações devem ser executadas em conjunto: verificação de estoque, processamento de pagamentos, confirmação do pedido e notificação de envio devem ocorrer dentro de uma janela de tempo específica para manter um estado consistente do pedido.

Impacto da modernização: pode exigir tratamento especial em sistemas distribuídos

Desafios:

- Coordenando dependências de tempo em serviços distribuídos

- Gerenciando transações distribuídas
- Confirmando a conclusão do processo em vários componentes

Melhores práticas para fortalecer a coesão:

- Implemente mecanismos de agendamento e tempos limite adequados
- Use arquiteturas orientadas por eventos com tratamento claro de sequências
- Design para uma eventual consistência com padrões de compensação
- Implemente padrões de saga para transações distribuídas

Padrão de coesão lógico ou coincidente

Definição: Os elementos são categorizados logicamente para fazer as mesmas coisas, mesmo que tenham relacionamentos fracos ou nenhum relacionamento significativo. Um exemplo é armazenar dados de pedidos de clientes, contagens de estoque de armazéns e modelos de e-mail de marketing no mesmo esquema de banco de dados, pois todos estão relacionados às operações de vendas, apesar de terem padrões de acesso, gerenciamento do ciclo de vida e requisitos de escalabilidade diferentes. Outro exemplo é combinar o processamento do pagamento de pedidos e o gerenciamento do catálogo de produtos no mesmo componente do banco de dados, pois ambos fazem parte do sistema de comércio eletrônico, embora atendam a funções comerciais distintas com necessidades operacionais diferentes.

Impacto da modernização: deve ser refatorado ou reorganizado

Desafios:

- Identificando melhores padrões organizacionais
- Quebrando dependências desnecessárias
- Componentes de reestruturação que foram agrupados arbitrariamente

Melhores práticas para fortalecer a coesão:

- Reorganize com base em verdadeiros limites funcionais e domínios de negócios
- Remova agrupamentos arbitrários com base em relacionamentos superficiais
- Implemente a separação adequada dos elementos com base nos recursos de negócios
- Alinhe os componentes do banco de dados com seus requisitos operacionais específicos

Implementando baixo acoplamento e alta coesão

Práticas recomendadas

As práticas recomendadas a seguir podem ajudá-lo a obter um baixo acoplamento:

- Minimize as dependências entre os componentes do banco de dados
- Use interfaces bem definidas para interação de componentes
- Evite estruturas de dados globais e estaduais compartilhadas

As melhores práticas a seguir podem ajudar você a alcançar uma alta coesão:

- Agrupe dados e operações relacionados
- Certifique-se de que cada componente tenha uma responsabilidade única e clara
- Mantenha limites claros entre diferentes domínios de negócios

Fase 1: Mapear dependências de dados

Mapeie relacionamentos de dados e identifique limites naturais. Você pode usar ferramentas, como SchemaSpy, para visualizar o banco de dados mostrando as tabelas no diagrama entidade-relacionamento (ER). Isso fornece uma análise estática do banco de dados e indica alguns dos limites e dependências claros dentro do banco de dados.

Você também pode exportar seus esquemas de banco de dados em um banco de dados gráfico ou em um Jupiter notebook. Em seguida, você pode aplicar algoritmos de agrupamento ou componentes interconectados para identificar limites e dependências naturais. Outras AWS Partner ferramentas, como CAST Imaging, podem ajudar a entender suas dependências de banco de dados.

Fase 2: Analise os limites das transações e os padrões de acesso

Analise os padrões de transação para manter as propriedades de atomicidade, consistência, isolamento e durabilidade (ACID) e entender como os dados são acessados e modificados. Você pode usar ferramentas de análise e diagnóstico de banco de dados, como Oracle Automatic Workload Repository (AWR)ou PostgreSQL pg_stat_statements. Essa análise ajuda você a entender quem está acessando o banco de dados e quais são os limites da transação. Também pode ajudá-lo a entender a coesão e o acoplamento entre as tabelas em tempo de execução. Você também pode

usar ferramentas de monitoramento e criação de perfil que podem vincular perfis de execução de código e banco de dados, como Dynatrace AppEngine.

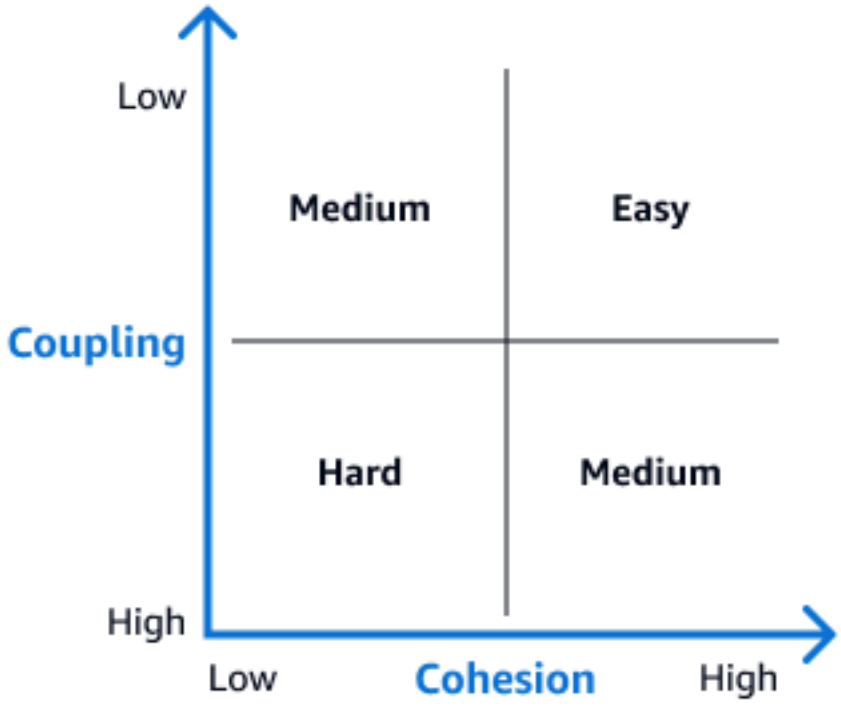
Ferramentas de IA, como vFunction, podem ajudá-lo a identificar limites de domínio analisando os limites funcionais e de domínio do aplicativo. Embora analise vFunction principalmente a camada do aplicativo, seus insights podem orientar a decomposição do aplicativo e do banco de dados, apoiando o alinhamento com os domínios de negócios.

Fase 3: Identificar tabelas independentes

Procure tabelas que demonstrem duas características principais:

- Alta coesão — Os conteúdos da tabela estão fortemente relacionados entre si
- Baixo acoplamento — Eles têm dependências mínimas em outras tabelas.

A seguinte matriz de acoplamento e coesão pode ajudá-lo a identificar a dificuldade de desacoplar cada tabela. As tabelas que aparecem no quadrante superior direito dessa matriz são candidatas ideais para os esforços iniciais de desacoplamento, pois são as mais fáceis de separar. Em um diagrama ER, essas tabelas têm poucos relacionamentos de chave estrangeira ou outras dependências. Depois de desacoplar essas tabelas, avance para tabelas com relacionamentos mais complexos.



> **Note**
>
> A estrutura do banco de dados geralmente reflete a arquitetura do aplicativo. Tabelas que são mais fáceis de desacoplar no nível do banco de dados normalmente correspondem a componentes que são mais fáceis de converter em microsserviços no nível do aplicativo.

Migração da lógica de negócios do banco de dados para a camada do aplicativo

A migração da lógica de negócios de procedimentos, acionadores e funções armazenados no banco de dados para serviços da camada de aplicativos é uma etapa essencial na decomposição de bancos de dados monolíticos. Essa transformação melhora a autonomia do serviço, simplifica a manutenção e melhora a escalabilidade. Esta seção fornece orientação sobre como analisar a lógica do banco de dados, planejar a estratégia de migração e, em seguida, implementar a transformação, mantendo a continuidade dos negócios. Também discute o estabelecimento de um plano de reversão eficaz.

Esta seção contém os seguintes tópicos:

Fase 1: Analisando a lógica de negócios

Ao modernizar bancos de dados monolíticos, você deve primeiro realizar uma análise abrangente da lógica de seu banco de dados existente. Essa fase se concentra em três categorias principais:

- Os procedimentos armazenados geralmente contêm operações comerciais críticas, incluindo lógica de manipulação de dados, regras de negócios, verificações de validação e cálculos. Como componentes principais da lógica de negócios do aplicativo, eles exigem uma decomposição cuidadosa. Por exemplo, os procedimentos armazenados de uma organização financeira podem lidar com cálculos de juros, reconciliação de contas e verificações de conformidade.
- Os acionadores são os principais componentes do banco de dados que lidam com trilhas de auditoria, validação de dados, cálculos e consistência entre tabelas. Por exemplo, uma organização de varejo pode usar gatilhos para gerenciar atualizações de estoque em todo o sistema de processamento de pedidos, o que demonstra a complexidade das operações automatizadas do banco de dados.
- As funções em bancos de dados gerenciam principalmente transformações de dados, cálculos e operações de pesquisa. Eles geralmente são incorporados em vários procedimentos e aplicativos.

Por exemplo, uma organização de saúde pode usar funções para normalizar os dados do paciente ou consultar códigos médicos.

Cada categoria representa aspectos diferentes da lógica de negócios que está incorporada na camada do banco de dados. Você precisa avaliar e planejar cuidadosamente cada uma delas para migrá-las para a camada de aplicação.

Durante essa fase de análise, os clientes normalmente enfrentam três desafios significativos. Primeiro, dependências complexas surgem por meio de chamadas de procedimentos aninhadas, referências entre esquemas e dependências de dados implícitas. Em segundo lugar, o gerenciamento de transações se torna essencial, principalmente ao lidar com transações em várias etapas e manter a consistência dos dados em sistemas distribuídos. Em terceiro lugar, as considerações de desempenho devem ser cuidadosamente avaliadas, especialmente para operações de processamento em lote, atualizações de dados em massa e cálculos em tempo real que atualmente se beneficiam da proximidade com os dados.

Para enfrentar esses desafios de forma eficaz, você pode usar AWS Schema Conversion Tool (AWS SCT) para análise inicial e, em seguida, usar ferramentas detalhadas de mapeamento de dependências. Essa abordagem ajuda você a entender o escopo completo da lógica do seu banco de dados e a criar uma estratégia de migração abrangente que mantenha a continuidade dos negócios durante a decomposição.

Ao entender completamente esses componentes e desafios, você pode planejar melhor sua jornada de modernização e tomar decisões informadas sobre quais elementos priorizar durante a migração para uma arquitetura baseada em microsserviços.

Ao analisar os componentes do código do banco de dados, crie uma documentação abrangente para cada procedimento, acionador e função armazenados. Comece descrevendo claramente sua finalidade e funcionalidade principal, incluindo as regras de negócios que ela implementa. Detalhe todos os parâmetros de entrada e saída e anote seus tipos de dados e intervalos válidos. Mapeie dependências em outros objetos de banco de dados, sistemas externos e processos posteriores. Defina claramente os limites da transação e os requisitos de isolamento para manter a integridade dos dados. Documente todas as expectativas de desempenho, incluindo requisitos de tempo de resposta e padrões de utilização de recursos. Por fim, analise os padrões de uso para entender os picos de carga, a frequência de execução e os períodos comerciais críticos.

Fase 2: Classificando a lógica de negócios

A decomposição eficaz do banco de dados requer a categorização sistemática da lógica do banco de dados em todas as principais dimensões: complexidade, impacto nos negócios, dependências, padrões de uso e dificuldade de migração. Essa classificação ajuda você a identificar componentes de alto risco, determinar os requisitos de teste e estabelecer prioridades de migração. Por exemplo, procedimentos armazenados complexos com alto impacto nos negócios e uso frequente exigem um planejamento cuidadoso e testes extensivos. No entanto, funções simples, raramente usadas, com dependências mínimas, podem ser adequadas para as fases iniciais da migração.

Essa abordagem estruturada cria um roteiro de migração equilibrado que minimiza a interrupção dos negócios e, ao mesmo tempo, mantém a estabilidade do sistema. Ao entender essas inter-relações, você pode melhorar a sequência de seus esforços de decomposição e alocar recursos de forma adequada.

Fase 3: Migração da lógica de negócios

Depois de analisar e classificar sua lógica de negócios, é hora de migrá-la. Há duas abordagens ao migrar a lógica de negócios de um banco de dados monolítico: mover a lógica do banco de dados para a camada do aplicativo ou mover a lógica comercial para outro banco de dados que faça parte do microsserviço.

Se você migrar a lógica de negócios para o aplicativo, as tabelas do banco de dados armazenarão somente os dados, e o banco de dados não conterá nenhuma lógica comercial. Essa é a abordagem recomendada. Você pode usar o Ispirer ou ferramentas generativas de IA, como o Amazon Q Developer ou Kiro, para converter a lógica de negócios do banco de dados para a camada do aplicativo, como a conversão para Java. Para obter mais informações, consulte Migrar a lógica de negócios do banco de dados para o aplicativo para acelerar a inovação e a flexibilidade (postagem AWS no blog).

Se você migrar a lógica de negócios para outro banco de dados, poderá usar AWS Schema Conversion Tool (AWS SCT) para converter esquemas de banco de dados e objetos de código existentes em seu banco de dados de destino. Ele oferece suporte a serviços de AWS banco de dados específicos, como Amazon DynamoDB, Amazon Aurora e Amazon Redshift. Ao fornecer um relatório de avaliação abrangente e recursos de conversão automatizada, AWS SCT ajuda a simplificar o processo de transição, permitindo que você se concentre na otimização de sua nova estrutura de banco de dados para melhorar o desempenho e a escalabilidade. À medida que você avança em seu projeto de modernização, AWS SCT pode lidar com conversões incrementais

para oferecer suporte a uma abordagem em fases, permitindo validar e ajustar cada etapa da transformação do banco de dados.

Estratégia de reversão para lógica de negócios

Dois aspectos críticos de qualquer estratégia de decomposição são manter a compatibilidade com versões anteriores e implementar procedimentos abrangentes de reversão. Esses elementos trabalham juntos para ajudar a proteger as operações durante o período de transição. Esta seção descreve como gerenciar a compatibilidade durante o processo de decomposição e estabelecer recursos eficazes de reversão de emergência que protejam contra possíveis problemas.

Mantenha a compatibilidade com versões anteriores

Durante a decomposição do banco de dados, manter a compatibilidade com versões anteriores é essencial para transições suaves. Mantenha os procedimentos de banco de dados existentes temporariamente em vigor enquanto implementa gradualmente novas funcionalidades. Use o controle de versão para rastrear todas as alterações e gerenciar várias versões do banco de dados simultaneamente. Planeje um período de coexistência prolongado em que os sistemas de origem e de destino devem operar de forma confiável. Isso dá tempo para testar e validar o novo sistema antes de retirar os componentes antigos. Essa abordagem minimiza a interrupção dos negócios e fornece uma rede de segurança para reversão, se necessário.

Plano de reversão de emergência

Uma estratégia abrangente de reversão é essencial para a decomposição segura do banco de dados. Implemente sinalizadores de recursos em seu código para controlar qual versão da lógica de negócios está ativa. Isso permite que você alterne instantaneamente entre as implementações nova e original sem alterações na implantação. Essa abordagem fornece controle refinado sobre a transição e ajuda você a reverter rapidamente se surgirem problemas. Mantenha a lógica original como um backup verificado e mantenha procedimentos detalhados de reversão que especificam acionadores, responsabilidades e etapas de recuperação.

Teste regularmente esses cenários de reversão sob várias condições para validar sua eficácia e garantir que as equipes estejam familiarizadas com os procedimentos de emergência. Os sinalizadores de recursos também permitem lançamentos graduais ao habilitar seletivamente novas funcionalidades para grupos de usuários ou transações específicas. Isso fornece uma camada adicional de mitigação de riscos durante a transição.

Desacoplando relacionamentos de tabelas durante a decomposição do banco de dados

Esta seção fornece orientação sobre como detalhar relacionamentos complexos de tabelas e operações JOIN durante a decomposição monolítica do banco de dados. Uma junção de tabela combina linhas de duas ou mais tabelas com base em uma coluna relacionada entre elas. O objetivo de separar esses relacionamentos é reduzir o alto acoplamento entre tabelas e, ao mesmo tempo, manter a integridade dos dados nos microsserviços.

Esta seção contém os seguintes tópicos:

Estratégia de desnormalização

A desnormalização é uma estratégia de design de banco de dados que envolve a introdução intencional de redundância combinando ou duplicando dados em tabelas. Ao dividir um banco de dados grande em bancos de dados pequenos, pode fazer sentido duplicar alguns dados entre os serviços. Por exemplo, armazenar detalhes básicos do cliente, como nome e endereço de e-mail, tanto em um serviço de marketing quanto em um serviço de pedidos elimina a necessidade de pesquisas constantes entre serviços. O serviço de marketing pode precisar das preferências do cliente e das informações de contato para direcionar campanhas, enquanto o serviço de pedidos exige os mesmos dados para processamento de pedidos e notificações. Embora isso crie alguma redundância de dados, pode melhorar significativamente o desempenho e a independência do serviço, permitindo que a equipe de marketing opere suas campanhas sem depender de consultas de atendimento ao cliente em tempo real.

Ao implementar a desnormalização, concentre-se nos campos acessados com frequência que você identifica por meio de uma análise cuidadosa dos padrões de acesso aos dados. Você pode usar ferramentas, como Oracle AWR relatórios oupg_stat_statements, para entender quais

dados geralmente são recuperados juntos. Os especialistas do domínio também podem fornecer informações valiosas sobre agrupamentos de dados naturais. Lembre-se de que a desnormalização não é uma all-or-nothing abordagem — apenas dados duplicados que comprovadamente melhoram o desempenho do sistema ou reduzem dependências complexas.

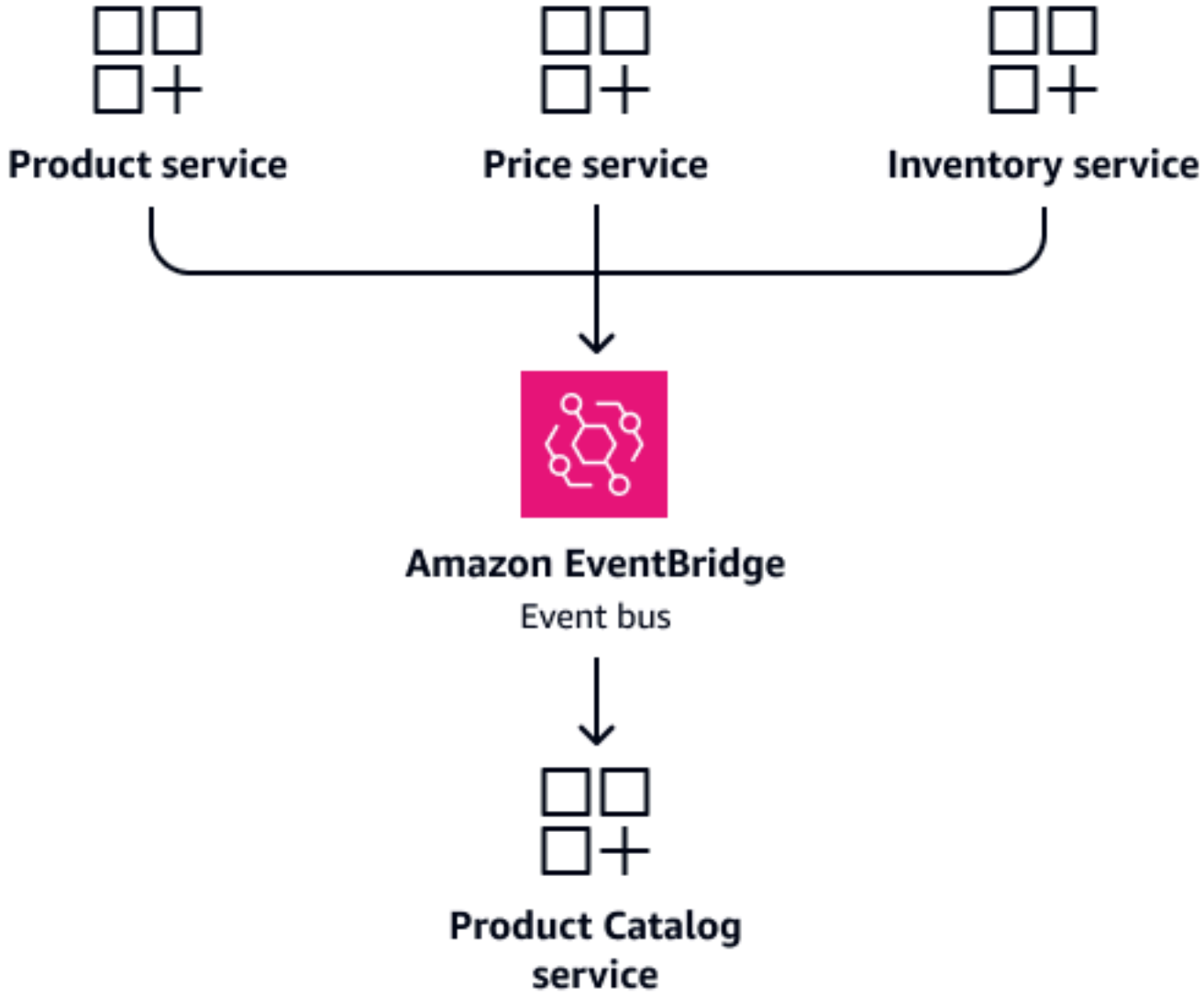
Reference-by-key estratégia

Uma reference-by-key estratégia é um padrão de design de banco de dados em que os relacionamentos entre entidades são mantidos por meio de chaves exclusivas, em vez de armazenar os dados reais relacionados. Em vez dos relacionamentos tradicionais de chave estrangeira, os microsserviços modernos geralmente armazenam apenas os identificadores exclusivos dos dados relacionados. Por exemplo, em vez de manter todos os detalhes do cliente na tabela de pedidos, o serviço de pedidos armazena apenas a ID do cliente e recupera informações adicionais do cliente por meio de uma chamada de API quando necessário. Essa abordagem mantém a independência do serviço e, ao mesmo tempo, garante o acesso aos dados relacionados.

Padrão CQRS

O padrão Command Query Responsibility Segregation (CQRS) separa as operações de leitura e gravação de um armazenamento de dados. Esse padrão é particularmente útil em sistemas complexos com requisitos de alto desempenho, especialmente aqueles com cargas assimétricas read/write . Se seu aplicativo frequentemente precisa de dados combinados de várias fontes, você pode criar um modelo CQRS dedicado em vez de junções complexas. Por exemplo, em vez de unir `Product` `Inventory` tabelas e em cada solicitação, mantenha uma `Product Catalog` tabela consolidada que contenha os dados necessários. `Pricing` Os benefícios dessa abordagem podem superar os custos da tabela adicional.

Considere um cenário em que`Product`,`Price`, e `Inventory` os serviços frequentemente precisam de informações sobre o produto. Em vez de configurar esses serviços para acessar diretamente as tabelas compartilhadas, crie um `Product Catalog` serviço dedicado. Esse serviço mantém seu próprio banco de dados que contém as informações consolidadas do produto. Ele atua como uma única fonte confiável para consultas relacionadas a produtos. Quando os detalhes do produto, os preços ou os níveis de estoque mudam, os respectivos serviços podem publicar eventos para atualizar o `Product Catalog` serviço. Isso fornece consistência de dados e, ao mesmo tempo, mantém a independência do serviço. A imagem a seguir mostra essa configuração, na qual a Amazon EventBridge serve como um barramento de eventos.



Conforme discutido na próxima seçãoSincronização de dados baseada em eventos, mantenha o modelo CQRS atualizado por meio de eventos. Quando os detalhes do produto, os preços ou os níveis de estoque mudam, os respectivos serviços publicam eventos. O `Product Catalog` serviço se inscreve nesses eventos e atualiza sua visão consolidada. Isso fornece leituras rápidas sem junções complexas e mantém a independência do serviço.

Sincronização de dados baseada em eventos

A sincronização de dados baseada em eventos é um padrão em que as alterações nos dados são capturadas e propagadas como eventos, o que permite que diferentes sistemas ou componentes mantenham estados de dados sincronizados. Quando os dados mudarem, em vez de atualizar todos os bancos de dados relacionados imediatamente, publique um evento para notificar os serviços

assinados. Por exemplo, quando um cliente altera seu endereço de entrega no `Customer` serviço, um `CustomerUpdated` evento inicia atualizações no `Order` serviço e no `Delivery` serviço de acordo com a programação de cada serviço. Essa abordagem substitui as uniões rígidas de tabelas por atualizações flexíveis e escaláveis orientadas por eventos. Alguns serviços podem ter dados desatualizados por um breve período, mas a desvantagem é melhorar a escalabilidade do sistema e a independência do serviço.

Implementando alternativas às junções de tabelas

Comece a decomposição do banco de dados com operações de leitura, pois elas geralmente são mais simples de migrar e validar. Depois que os caminhos de leitura estiverem estáveis, realize as operações de gravação mais complexas. Para requisitos críticos de alto desempenho, considere implementar o padrão CQRS. Use um banco de dados separado e otimizado para leituras e mantenha outro para gravações.

Crie sistemas resilientes adicionando lógica de repetição para chamadas entre serviços e implementando camadas de cache apropriadas. Monitore de perto as interações de serviço e configure alertas para problemas de consistência de dados. O objetivo final não é a consistência perfeita em todos os lugares — é criar serviços independentes que tenham um bom desempenho e, ao mesmo tempo, mantenham uma precisão de dados aceitável para suas necessidades comerciais.

A natureza desacoplada dos microsserviços introduz as seguintes novas complexidades no gerenciamento de dados:

- Os dados são distribuídos. Os dados agora residem em bancos de dados separados, que são gerenciados por serviços independentes.
- A sincronização em tempo real entre serviços geralmente é impraticável, exigindo um modelo de consistência eventual.
- As operações que antes ocorriam em uma única transação de banco de dados agora abrangem vários serviços.

Para enfrentar esses desafios, faça o seguinte:

- Implemente uma arquitetura orientada por eventos — use filas de mensagens e publicação de eventos para propagar as alterações de dados entre os serviços. Para obter mais informações, consulte Criação de arquiteturas orientadas a eventos em terrenos sem servidor.

- Adote o padrão de orquestração da saga — Esse padrão ajuda você a gerenciar transações distribuídas e manter a integridade dos dados em todos os serviços. Para obter mais informações, consulte Criação de um aplicativo distribuído sem servidor usando um padrão de orquestração de saga em blogs. AWS
- Projete para falhas — incorpore mecanismos de repetição, disjuntores e transações de compensação para lidar com problemas de rede ou falhas de serviço.
- Use o carimbo de versão — Acompanhe as versões de dados para gerenciar conflitos e garantir que as atualizações mais recentes sejam aplicadas.
- Reconciliação regular — implemente processos periódicos de sincronização de dados para capturar e corrigir quaisquer inconsistências.

Exemplo baseado em cenários

O exemplo de esquema a seguir tem duas tabelas, uma `Customer` tabela e uma `Order` tabela:

```
-- Customer table
CREATE TABLE customer (
    customer_id INT PRIMARY KEY,
    first_name VARCHAR(100),
    last_name VARCHAR(100),
    email VARCHAR(255),
    phone VARCHAR(20),
    address TEXT,
    created_at TIMESTAMP
);

-- Order table
CREATE TABLE order (
    order_id INT PRIMARY KEY,
    customer_id INT,
    order_date TIMESTAMP,
    total_amount DECIMAL(10,2),
    status VARCHAR(50),
    FOREIGN KEY (customer_id) REFERENCES customers(id)
);
```

Veja a seguir um exemplo de como você pode usar uma abordagem desnormalizada:

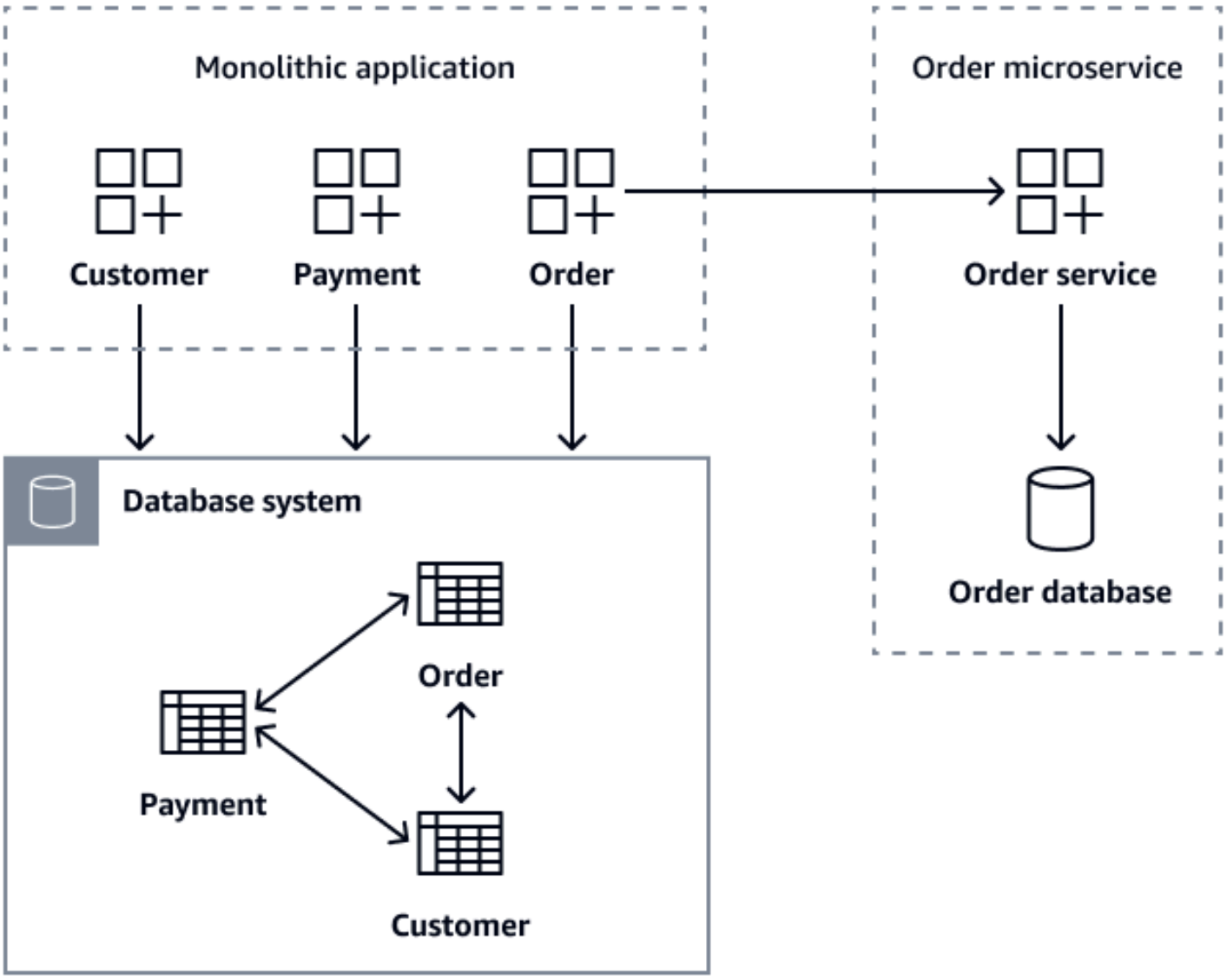
```
CREATE TABLE order (
```

```
    order_id INT PRIMARY KEY,
    customer_id INT,                  -- Reference only
    customer_first_name VARCHAR(100), -- Denormalized
    customer_last_name VARCHAR(100),  -- Denormalized
    customer_email VARCHAR(255),      -- Denormalized
    order_date TIMESTAMP,
    total_amount DECIMAL(10,2),
    status VARCHAR(50)
);
```

A nova `Order` tabela tem o nome do cliente e os endereços de e-mail que estão desnormalizados. O `customer_id` é referenciado e não há nenhuma restrição de chave estrangeira com a `Customer` tabela. A seguir estão os benefícios dessa abordagem desnormalizada:

- O `Order` serviço pode exibir o histórico de pedidos com detalhes do cliente e não exige chamadas de API para o `Customer` microsserviço.
- Se o `Customer` serviço estiver inativo, ele `Order` permanecerá totalmente funcional.
- As consultas para processamento e geração de relatórios de pedidos são executadas mais rapidamente.

O diagrama a seguir mostra um aplicativo monolítico que recupera dados de pedidos usando`getOrder(customer_id)`, `getOrder(order_id)``getCustomerOders(customer_id)`, e chamadas de `createOrder(Order order)` API para o microsserviço. `Order`



Durante a migração de microsserviços, você pode manter a `Order` tabela no banco de dados monolítico como uma medida de segurança transitória, garantindo que o aplicativo legado permaneça funcional. No entanto, é fundamental que todas as novas operações relacionadas a pedidos sejam roteadas por meio da API de `Order` microsserviços, que mantém seu próprio banco de dados e, ao mesmo tempo, grava no banco de dados legado como backup. Esse padrão de gravação dupla fornece uma rede de segurança. Ele permite a migração gradual enquanto mantém a estabilidade do sistema. Depois que todos os clientes tiverem migrado com sucesso para o novo microsserviço, você poderá descontinuar a `Order` tabela legada no banco de dados monolítico. Depois de decompor o aplicativo monolítico e seu banco de dados em `Order` microsserviços separados`Customer`, manter a consistência dos dados se torna o principal desafio.

Práticas recomendadas para decomposição de banco de dados

Ao decompor um banco de dados monolítico, as organizações devem estabelecer estruturas claras para monitorar o progresso, manter o conhecimento do sistema e enfrentar os desafios emergentes. Esta seção fornece as melhores práticas para medir o sucesso da decomposição, manter a documentação crucial, implementar processos de melhoria contínua e enfrentar desafios comuns. Compreender e seguir essas diretrizes ajuda a garantir que os esforços de decomposição do banco de dados forneçam os benefícios pretendidos e, ao mesmo tempo, minimizem as interrupções operacionais e o débito técnico.

Esta seção contém os seguintes tópicos:

Medindo o sucesso

Acompanhe o sucesso da decomposição por meio de uma combinação de métricas técnicas, operacionais e comerciais. Tecnicamente, monitore os tempos de resposta das consultas, as melhorias no tempo de atividade do sistema e os aumentos na frequência de implantação. Operacionalmente, meça as reduções de incidentes, a velocidade de resolução de problemas e as melhorias na utilização de recursos. Para desenvolvimento, monitore a velocidade de implementação de recursos, a aceleração do ciclo de lançamento e a redução nas dependências entre equipes. Os impactos nos negócios devem resultar em custos operacionais reduzidos, maior rapidez time-to-market e maior satisfação do cliente. Essas métricas geralmente são definidas durante a fase de escopo. Para obter mais informações, consulte Definindo o escopo e os requisitos para a decomposição do banco de dados neste guia.

Requisitos de documentação

Mantenha a documentação da arquitetura up-to-date do sistema com limites de serviço, fluxos de dados e especificações de interface claros. Use registros de decisão de arquitetura (ADRs) para

capturar as principais decisões técnicas, incluindo seu contexto, consequências e alternativas consideradas. Por exemplo, documente por que serviços específicos foram separados primeiro ou como certas compensações de consistência de dados foram feitas.

Agende análises mensais da arquitetura para avaliar a integridade do sistema por meio das principais métricas: tendências de desempenho, conformidade de segurança e dependências entre serviços. Inclua feedback das equipes de desenvolvimento sobre desafios de integração e problemas operacionais. Esse ciclo regular de revisão ajuda a identificar problemas emergentes com antecedência e valida que os esforços de decomposição permaneçam alinhados às metas de negócios.

Estratégia de melhoria contínua

Trate a decomposição do banco de dados como um processo iterativo, não como um projeto único. Monitore as métricas de desempenho do sistema e as interações de serviço para identificar oportunidades de otimização. A cada trimestre, priorize o tratamento da dívida técnica com base no impacto operacional e nos custos de manutenção. Por exemplo, automatize operações de banco de dados realizadas com frequência, aprimore a cobertura de monitoramento e refine os procedimentos de implantação com base nos padrões aprendidos.

Superando desafios comuns na decomposição de bancos de dados

A otimização do desempenho requer uma abordagem multifacetada. Implemente o armazenamento em cache estratégico nos limites do serviço, otimize os padrões de consulta com base no uso real e monitore continuamente as principais métricas. Aborde os gargalos de desempenho de forma proativa analisando tendências e definindo limites claros para intervenção.

Os desafios de consistência de dados exigem escolhas arquitetônicas cuidadosas. Implemente padrões orientados por eventos para atualizações entre serviços e use padrões de orquestração de saga para transações complexas. Defina limites claros de serviço e aceite a consistência eventual quando os requisitos de negócios permitirem. Esse equilíbrio entre consistência e autonomia de serviço é crucial para uma decomposição bem-sucedida.

A excelência operacional exige automação de tarefas rotineiras e procedimentos padronizados em todos os serviços. Mantenha um monitoramento abrangente com limites claros de alerta e invista no treinamento regular da equipe para novos padrões e ferramentas. Essa abordagem sistemática das operações promove a prestação confiável de serviços e, ao mesmo tempo, gerencia a complexidade.

Perguntas frequentes sobre decomposição de banco de dados

Esta seção abrangente de perguntas frequentes aborda as perguntas e desafios mais comuns que as organizações enfrentam ao realizar projetos de decomposição de banco de dados. Desde a definição do escopo e dos requisitos iniciais até a migração de procedimentos armazenados, essas perguntas fornecem insights práticos e abordagens estratégicas para ajudar as equipes a percorrerem com sucesso sua jornada de modernização do banco de dados. Se você está na fase de planejamento ou já está executando sua estratégia de decomposição, essas respostas podem ajudá-lo a evitar armadilhas comuns e implementar as melhores práticas para obter os melhores resultados.

Esta seção contém os seguintes tópicos:

- FAQs sobre a definição do escopo e dos requisitos
- FAQs sobre o controle do acesso ao banco de dados
- FAQs sobre análise de coesão e acoplamento
- FAQs sobre a migração da lógica de negócios para a camada de aplicação

FAQs sobre a definição do escopo e dos requisitos

A Definindo o escopo e os requisitos para a decomposição do banco de dados seção deste guia discute como analisar interações, mapear dependências e estabelecer critérios de sucesso. Esta seção de perguntas frequentes aborda as principais questões sobre como estabelecer e gerenciar os limites do projeto. Se você está lidando com restrições técnicas pouco claras, necessidades departamentais conflitantes ou requisitos comerciais em evolução, eles FAQs fornecem orientação prática sobre como manter uma abordagem equilibrada.

Esta seção contém as seguintes perguntas:

- Quão detalhada deve ser a definição inicial do escopo?
- E se eu descobrir dependências adicionais depois de iniciar o projeto?
- Como faço para lidar com partes interessadas de diferentes departamentos que têm requisitos conflitantes?
- Qual é a melhor maneira de avaliar as restrições técnicas quando a documentação é deficiente ou desatualizada?

- Como faço para equilibrar as necessidades comerciais imediatas com as metas técnicas de longo prazo?
- Como posso ter certeza de que não estou perdendo requisitos críticos de partes interessadas silenciosas?
- Essas recomendações se aplicam a bancos de dados de mainframe monolíticos?

Quão detalhada deve ser a definição inicial do escopo?

Trabalhando de acordo com as necessidades de seus clientes, defina o escopo do projeto com detalhes suficientes para identificar os limites do sistema e as dependências críticas, mantendo a flexibilidade para a descoberta. Mapeie elementos essenciais, incluindo interfaces de sistema, principais partes interessadas e principais restrições técnicas. Comece aos poucos, selecionando uma parte limitada e de baixo risco do sistema que forneça valor mensurável. Essa abordagem ajuda as equipes a aprender e ajustar estratégias antes de lidar com componentes mais complexos.

Documente os requisitos comerciais essenciais que impulsionam o esforço de decomposição, mas evite especificar demais os detalhes que podem mudar durante a implementação. Essa abordagem equilibrada garante que as equipes possam avançar com clareza e, ao mesmo tempo, permanecer adaptáveis aos novos insights e desafios que surgem durante a jornada de modernização.

E se eu descobrir dependências adicionais depois de iniciar o projeto?

Espere descobrir dependências adicionais à medida que o projeto progride. Mantenha um registro de dependências ativo e realize revisões regulares do escopo para avaliar o impacto nos cronogramas e nos recursos. Implemente um processo claro de gerenciamento de mudanças e inclua tempo de reserva nos planos do projeto para lidar com descobertas inesperadas. O objetivo não é evitar mudanças, mas gerenciá-las de forma eficaz. Isso ajuda as equipes a se adaptarem rapidamente, mantendo a dinâmica do projeto.

Como faço para lidar com partes interessadas de diferentes departamentos que têm requisitos conflitantes?

Lide com requisitos departamentais conflitantes por meio de uma priorização clara baseada no valor comercial e no impacto do sistema. Garanta o patrocínio executivo para tomar decisões importantes e resolver conflitos rapidamente. Agende reuniões regulares de alinhamento das partes interessadas para discutir compensações e manter a transparência. Documente todas as decisões e

seus fundamentos para promover uma comunicação clara e manter a dinâmica do projeto. Concentre as discussões nos benefícios comerciais quantificáveis, em vez das preferências departamentais.

Qual é a melhor maneira de avaliar as restrições técnicas quando a documentação é deficiente ou desatualizada?

Ao enfrentar uma documentação deficiente, combine a análise tradicional com ferramentas modernas de IA. Use modelos de linguagem grandes (LLMs) para analisar repositórios de código, registros e documentação existente a fim de identificar padrões e possíveis restrições. Entreviste desenvolvedores experientes e arquitetos de banco de dados para validar as descobertas da IA e descobrir restrições não documentadas. Implemente ferramentas de monitoramento que tenham recursos aprimorados de IA para observar o comportamento do sistema e prever possíveis problemas.

Crie pequenos experimentos técnicos que validem suas suposições. Você pode usar ferramentas de teste baseadas em IA para acelerar o processo. Documente as descobertas em uma base de conhecimento que pode ser continuamente aprimorada por meio de atualizações assistidas por IA. Considere contratar especialistas no assunto para áreas complexas e usar ferramentas de programação em pares de IA para acelerar seus esforços de análise e documentação.

Como faço para equilibrar as necessidades comerciais imediatas com as metas técnicas de longo prazo?

Crie um roteiro de projeto em fases que alinhe as necessidades comerciais imediatas com os objetivos técnicos de longo prazo. Identifique ganhos rápidos que ofereçam valor tangível desde o início para que você possa aumentar a confiança das partes interessadas. Divida a decomposição em marcos claros. Cada um deve oferecer benefícios comerciais mensuráveis e, ao mesmo tempo, progredir em direção às metas arquitetônicas. Mantenha a flexibilidade para atender às necessidades urgentes dos negócios por meio de revisões e ajustes regulares do roteiro.

Como posso ter certeza de que não estou perdendo requisitos críticos de partes interessadas silenciosas?

Mapeie todas as partes interessadas em potencial em toda a organização, incluindo proprietários de sistemas posteriores e usuários indiretos. Crie vários canais de feedback por meio de entrevistas estruturadas, workshops e sessões regulares de revisão. Crie proof-of-concepts e crie protótipos para tornar os requisitos tangíveis e estimular discussões significativas. Por exemplo, um painel

simples que mostra as dependências do sistema geralmente revela partes interessadas e requisitos ocultos que inicialmente não eram aparentes.

Conduza sessões regulares de validação com partes interessadas que falam e se certifique de que todas as perspectivas sejam capturadas. Os insights críticos geralmente vêm das pessoas mais próximas das operações diárias, e não das vozes mais altas nas reuniões de planejamento.

Essas recomendações se aplicam a bancos de dados de mainframe monolíticos?

A metodologia descrita neste guia também se aplica à decomposição de bancos de dados de mainframe monolíticos. Os principais desafios desses bancos de dados são gerenciar os requisitos das várias partes interessadas. As recomendações de tecnologia neste guia podem se aplicar a bancos de dados de mainframe monolíticos. Se o mainframe tiver um banco de dados relacional, como um banco de dados de processamento de transações on-line (OLTP), muitas das recomendações serão aplicadas. Para bancos de dados de processamento analítico on-line (OLAP), como aqueles usados para gerar relatórios comerciais, somente algumas das recomendações se aplicam.

FAQs sobre o controle do acesso ao banco de dados

O controle do acesso ao banco de dados usando o padrão de serviço do wrapper do banco de dados é discutido na Controle do acesso ao banco de dados durante a decomposição seção deste guia. Esta seção de perguntas frequentes aborda preocupações e dúvidas comuns sobre a introdução de um serviço de encapsulamento de banco de dados, incluindo seu impacto potencial no desempenho, no tratamento de procedimentos armazenados existentes, no gerenciamento de transações complexas e na supervisão de alterações no esquema.

Esta seção contém as seguintes perguntas:

- O serviço de embalagem não se tornará um novo gargalo?
- O que acontece com os procedimentos armazenados existentes?
- Como faço para gerenciar as mudanças no esquema durante a transição?

O serviço de embalagem não se tornará um novo gargalo?

Embora o serviço de encapsulamento de banco de dados adicione um salto extra na rede, o impacto geralmente é mínimo. Você pode escalar o serviço horizontalmente, e os benefícios do acesso

controlado geralmente superam o pequeno custo de desempenho. Considere isso uma troca temporária entre desempenho e capacidade de manutenção.

O que acontece com os procedimentos armazenados existentes?

Inicialmente, o serviço de encapsulamento de banco de dados pode expor procedimentos armazenados como métodos de serviço. Com o tempo, você pode mover gradualmente a lógica para a camada do aplicativo, o que melhora os testes e o controle de versão. Migre a lógica de negócios de forma incremental para minimizar os riscos.

Como faço para gerenciar as mudanças no esquema durante a transição?

Centralize o controle de alterações de esquema por meio da equipe de serviço de empacotamento. Essa equipe é responsável por manter uma visibilidade abrangente de todos os consumidores. Essa equipe analisa as mudanças propostas para impactar todo o sistema, coordena com as equipes afetadas e implementa as modificações usando um processo de implantação controlado. Por exemplo, ao adicionar novos campos, essa equipe deve manter a compatibilidade com versões anteriores implementando valores padrão ou permitindo inicialmente nulos.

Estabeleça um processo claro de gerenciamento de mudanças que inclua avaliação de impacto, requisitos de teste e procedimentos de reversão. Use ferramentas de controle de versão do banco de dados e mantenha uma documentação clara de todas as alterações. Essa abordagem centralizada evita que as modificações do esquema interrompam os serviços dependentes e mantém a estabilidade do sistema.

FAQs sobre análise de coesão e acoplamento

Compreender e analisar com eficácia o acoplamento e a coesão do banco de dados é fundamental para uma decomposição bem-sucedida do banco de dados. O acoplamento e a coesão são discutidos na Analisando a coesão e o acoplamento para decomposição do banco de dados seção deste guia. Esta seção de perguntas frequentes aborda as principais questões sobre como identificar os níveis adequados de granularidade, selecionar as ferramentas de análise corretas, documentar descobertas e priorizar problemas de acoplamento.

Esta seção contém as seguintes perguntas:

- Como identifico o nível certo de granularidade ao analisar o acoplamento?
- Quais ferramentas posso usar para analisar o acoplamento e a coesão do banco de dados?

- Qual é a melhor maneira de documentar as descobertas de acoplamento e coesão?
- Como priorizo quais problemas de acoplamento devem ser resolvidos primeiro?
- Como faço para lidar com transações que abrangem várias operações?

Como identifico o nível certo de granularidade ao analisar o acoplamento?

Comece com uma ampla análise dos relacionamentos do banco de dados e, em seguida, faça um detalhamento sistemático para identificar os pontos de separação natural. Use ferramentas de análise de banco de dados para mapear relacionamentos em nível de tabela, dependências de esquema e limites de transações. Por exemplo, examine os padrões de junção em consultas SQL para entender as dependências de acesso aos dados. Você também pode analisar os registros de transações para identificar os limites do processo de negócios.

Concentre-se em áreas onde o acoplamento é naturalmente mínimo. Eles geralmente se alinham aos limites do domínio de negócios e representam pontos de decomposição ideais. Ao determinar os limites de serviço apropriados, considere tanto o acoplamento técnico (como tabelas compartilhadas e chaves estrangeiras) quanto o acoplamento comercial (como fluxos de processos e necessidades de geração de relatórios).

Quais ferramentas posso usar para analisar o acoplamento e a coesão do banco de dados?

Você pode usar uma combinação de ferramentas automatizadas e análise manual para avaliar o acoplamento e a coesão do banco de dados. As ferramentas a seguir podem ajudá-lo nessa avaliação:

- Ferramentas de visualização de esquema — Você pode usar ferramentas como SchemaSpyou pgAdminpara gerar diagramas ER. Esses diagramas revelam relações de tabela e possíveis pontos de acoplamento.
- Ferramentas de análise de consultas — Você pode usar pg_stat_statementsou SQL Server Query Storeidentificar tabelas e padrões de acesso associados com frequência.
- Ferramentas de criação de perfil de banco de dados — ferramentas como Oracle SQL Developerou MySQL Workbenchfornecem informações sobre desempenho de consultas e dependências de dados.
- Ferramentas de mapeamento de dependências — O AWS Schema Conversion Tool (AWS SCT) pode ajudá-lo a visualizar relacionamentos de esquema e identificar componentes fortemente

acoplados. vFunctionpode ajudá-lo a identificar limites de domínio analisando os limites funcionais e de domínio do aplicativo.

- Ferramentas de monitoramento de transações — Você pode usar ferramentas específicas do banco de dados, como Oracle Enterprise Managerou SQL Server Extended Events, para analisar os limites das transações.
- Ferramentas de migração de lógica de negócios — Você pode usar Ispirerferramentas de IA generativas, como Amazon Q Developer ou Kiro, para converter a lógica de negócios do banco de dados para a camada do aplicativo, como a conversão para Java.

Combine essas análises automatizadas com a revisão manual dos processos de negócios e o conhecimento do domínio para entender completamente o acoplamento do sistema. Essa abordagem multifacetada garante que as perspectivas técnicas e comerciais sejam consideradas em sua estratégia de decomposição.

Qual é a melhor maneira de documentar as descobertas de acoplamento e coesão?

Crie uma documentação abrangente que visualize os relacionamentos do banco de dados e os padrões de uso. A seguir estão os tipos de ativos que você pode usar para registrar suas descobertas:

- Matrizes de dependência — Mapeie as dependências da tabela e destaque as áreas de alto acoplamento.
- Diagramas de relacionamento — Use diagramas ER para mostrar conexões de esquema e relacionamentos de chave estrangeira.
- Mapas de calor de uso da tabela — Visualize a frequência das consultas e os padrões de acesso aos dados nas tabelas.
- Diagramas de fluxo de transações — documente transações de várias tabelas e seus limites.
- Mapas de limites de domínio — descreva possíveis limites de serviço com base nos domínios de negócios.

Combine esses artefatos em um documento e atualize-o regularmente à medida que a decomposição progride. Para diagramas, você pode usar ferramentas como draw.ioou Lucidchart. Considere implementar um wiki para facilitar o acesso e a colaboração da equipe. Essa abordagem

de documentação multifacetada fornece uma compreensão clara e compartilhada do acoplamento e coesão do sistema.

Como priorizo quais problemas de acoplamento devem ser resolvidos primeiro?

Priorize os problemas de acoplamento com base em uma avaliação equilibrada dos fatores comerciais e técnicos. Avalie cada problema em relação ao impacto nos negócios (como receita e experiência do cliente), risco técnico (como estabilidade do sistema e integridade dos dados), esforço de implementação e recursos da equipe. Crie uma matriz de priorização que pontua cada problema de 1 a 5 nessas dimensões. Essa matriz ajuda você a identificar as oportunidades mais valiosas com riscos gerenciáveis.

Comece com mudanças de alto impacto e baixo risco que se alinham à experiência existente da equipe. Isso ajuda você a criar confiança organizacional e impulso para mudanças mais complexas. Essa abordagem promove uma execução realista e maximiza o valor comercial. Analise e ajuste regularmente as prioridades para ajudar a manter o alinhamento com as mudanças nas necessidades de negócios e na capacidade da equipe.

Como faço para lidar com transações que abrangem várias operações?

Gerencie transações multioperacionais por meio de uma coordenação de nível de serviço cuidadosamente projetada. Implemente padrões de saga para transações distribuídas complexas. Divida-os em etapas menores e reversíveis que podem ser gerenciadas de forma independente. Por exemplo, um fluxo de processamento de pedidos pode ser dividido em etapas separadas para verificação de estoque, processamento de pagamentos e criação de pedidos, cada uma com seu próprio mecanismo de compensação.

Sempre que possível, redesenhe as operações para serem mais atômicas, o que reduz a necessidade de transações distribuídas. Quando as transações distribuídas forem inevitáveis, implemente mecanismos robustos de rastreamento e compensação para promover a consistência dos dados. Monitore as taxas de conclusão de transações e implemente procedimentos claros de recuperação de erros para manter a confiabilidade do sistema.

FAQs sobre a migração da lógica de negócios para a camada de aplicação

Migrar a lógica de negócios do banco de dados para a camada de aplicativos é um aspecto crítico e complexo da modernização do banco de dados. Essa migração da lógica de negócios é discutida na Migração da lógica de negócios do banco de dados para a camada do aplicativo seção deste guia. Esta seção de perguntas frequentes aborda perguntas comuns sobre como gerenciar essa transição de forma eficaz, desde a seleção dos candidatos iniciais para migração até o tratamento de procedimentos e gatilhos armazenados complexos.

Esta seção contém as seguintes perguntas:

- Como identifico quais procedimentos armazenados devem ser migrados primeiro?
- Quais são os riscos de mover a lógica para a camada de aplicação?
- Como faço para manter o desempenho ao afastar a lógica do banco de dados?
- O que devo fazer com procedimentos armazenados complexos que envolvem várias tabelas?
- Como faço para lidar com os acionadores do banco de dados durante a migração?
- Qual é a melhor maneira de testar a lógica de negócios migrada?
- Como gerencio o período de transição quando a lógica do banco de dados e do aplicativo existe?
- Como faço para lidar com cenários de erro na camada de aplicação que foram gerenciados anteriormente pelo banco de dados?

Como identifico quais procedimentos armazenados devem ser migrados primeiro?

Comece identificando procedimentos armazenados que oferecem a melhor combinação de baixo risco e alto valor de aprendizado. Concentre-se em procedimentos que tenham dependências mínimas, funcionalidade clara e impacto não crítico nos negócios. Eles são candidatos ideais para a migração inicial porque ajudam a equipe a criar confiança e estabelecer padrões. Por exemplo, escolha procedimentos que lidem com operações de dados simples em vez daqueles que gerenciam transações complexas ou lógica comercial crítica.

Use ferramentas de monitoramento de banco de dados para analisar padrões de uso e identificar procedimentos pouco acessados como candidatos iniciais. Essa abordagem minimiza os riscos comerciais e, ao mesmo tempo, fornece uma experiência valiosa para lidar com migrações mais

complexas posteriormente. Classifique cada procedimento em termos de complexidade, importância comercial e níveis de dependência para criar uma sequência de migração priorizada.

Quais são os riscos de mover a lógica para a camada de aplicação?

Mover a lógica do banco de dados para a camada de aplicação apresenta vários desafios importantes. O desempenho do sistema pode diminuir devido ao aumento das chamadas de rede, especialmente para operações com uso intenso de dados que antes eram tratadas no banco de dados. O gerenciamento de transações se torna mais complexo e exige uma coordenação cuidadosa para manter a integridade dos dados em todas as operações distribuídas. Garantir a consistência dos dados se torna um desafio, especialmente para operações que antes dependiam de restrições no nível do banco de dados.

A possível interrupção dos negócios durante a migração e a curva de aprendizado dos desenvolvedores também são preocupações significativas. Reduza esses riscos por meio de um planejamento completo, testes extensivos em ambientes escalonados e migração gradual que começa com componentes menos críticos. Implemente procedimentos robustos de monitoramento e reversão para identificar e resolver rapidamente os problemas na produção.

Como faço para manter o desempenho ao afastar a lógica do banco de dados?

Implemente mecanismos de armazenamento em cache apropriados para dados acessados com frequência, otimize os padrões de acesso aos dados para minimizar as chamadas de rede e use o processamento em lote para operações em massa. Para non-time-critical operações, considere o processamento assíncrono para melhorar a capacidade de resposta do sistema.

Monitore de perto as métricas de desempenho do aplicativo e ajuste-as conforme necessário. Por exemplo, você pode substituir várias operações de linha única pelo processamento em massa, armazenar em cache dados de referência que mudam com pouca frequência e otimizar os padrões de consulta para reduzir a transferência de dados. Testes e ajustes regulares de desempenho ajudam o sistema a manter tempos de resposta aceitáveis e melhoram a capacidade de manutenção e a escalabilidade.

O que devo fazer com procedimentos armazenados complexos que envolvem várias tabelas?

Aborde procedimentos armazenados complexos e com várias tabelas por meio da decomposição sistemática. Comece dividindo-os em componentes menores e logicamente coerentes e identifique

limites claros de transações e dependências de dados. Crie interfaces de serviço para cada componente lógico. Isso ajuda você a migrar gradualmente sem interromper a funcionalidade existente.

Implemente uma step-by-step migração, começando com os componentes menos acoplados. Para procedimentos altamente complexos, considere mantê-los temporariamente no banco de dados enquanto migra partes mais simples. Essa abordagem híbrida mantém a estabilidade do sistema enquanto você avança em direção aos seus objetivos arquitetônicos. Monitore continuamente o desempenho e a funcionalidade durante a migração e esteja preparado para ajustar sua estratégia com base nos resultados.

Como faço para lidar com os acionadores do banco de dados durante a migração?

Transforme os acionadores do banco de dados em manipuladores de eventos no nível do aplicativo enquanto mantém a funcionalidade do sistema. Substitua os acionadores síncronos por padrões orientados por eventos que enfileiram mensagens para operações assíncronas. Considere usar o Amazon Simple Notification Service (Amazon SNS) ou o Amazon Simple Queue Service (Amazon SQS) para as filas de mensagens. Para requisitos de auditoria, implemente o registro em nível de aplicativo ou use os recursos de captura de dados de alteração do banco de dados (CDC).

Analise o propósito e a criticidade de cada gatilho. Alguns acionadores podem ser melhor atendidos pela lógica do aplicativo, e outros podem exigir padrões de fornecimento de eventos para manter a consistência dos dados. Comece com acionadores simples, como registros de auditoria, antes de abordar os mais complexos que gerenciam as regras de negócios ou a integridade dos dados. Monitore cuidadosamente durante a migração para garantir que não haja perda de funcionalidade ou consistência de dados.

Qual é a melhor maneira de testar a lógica de negócios migrada?

Implemente uma abordagem de teste em várias camadas antes de implantar a lógica de negócios migrada. Comece com testes de unidade para o novo código do aplicativo e, em seguida, adicione testes de integração que cubram os fluxos end-to-end de negócios. Execute implementações antigas e novas em paralelo e, em seguida, compare os resultados para validar a equivalência funcional. Realize testes de desempenho sob várias condições de carga para verificar se o comportamento do sistema corresponde ou excede os recursos anteriores.

Use sinalizadores de recursos para controlar a implantação para que você possa reverter rapidamente se surgirem problemas. Envolva os usuários corporativos na validação, especialmente

para fluxos de trabalho críticos. Monitore as principais métricas durante a implantação inicial e aumente gradualmente o tráfego para a nova implementação. Durante todo o processo, mantenha a capacidade de reverter para a lógica original do banco de dados, se necessário.

Como gerencio o período de transição quando a lógica do banco de dados e do aplicativo existe?

Quando a lógica do banco de dados e do aplicativo estiverem em uso, implemente sinalizadores de recursos que controlem o fluxo de tráfego e permitam a troca rápida entre implementações antigas e novas. Mantenha um controle de versão rigoroso e documente claramente as implementações e suas respectivas responsabilidades. Configure um monitoramento abrangente para ambos os sistemas para identificar rapidamente quaisquer discrepâncias ou problemas de desempenho.

Estabeleça procedimentos claros de reversão para cada componente migrado para que você possa reverter para a lógica original, se necessário. Comunique-se regularmente com todas as partes interessadas sobre o status da transição, possíveis impactos e procedimentos de escalonamento. Essa abordagem ajuda você a migrar gradualmente, mantendo a estabilidade do sistema e a confiança das partes interessadas.

Como faço para lidar com cenários de erro na camada de aplicação que foram gerenciados anteriormente pelo banco de dados?

Substitua o tratamento de erros no nível do banco de dados por mecanismos robustos na camada de aplicativos. Implemente disjuntores e repita a lógica para falhas transitórias. Use transações de compensação para manter a consistência dos dados em operações distribuídas. Por exemplo, se uma atualização de pagamento falhar, o aplicativo deverá tentar novamente automaticamente dentro dos limites definidos e iniciar ações de compensação, se necessário.

Configure monitoramento e alertas abrangentes para identificar problemas rapidamente e manter registros de auditoria detalhados para solução de problemas. Projete o tratamento de erros para ser o mais automatizado possível e defina caminhos claros de escalonamento para cenários que exijam intervenção humana. Essa abordagem em várias camadas fornece resiliência ao sistema enquanto mantém a integridade dos dados e a continuidade dos processos de negócios.

Próximas etapas para a decomposição do banco de dados em AWS

Depois de implementar as estratégias iniciais de decomposição do banco de dados por meio de serviços de encapsulamento de banco de dados e mover a lógica de negócios para a camada de aplicativos, as organizações devem planejar sua próxima evolução. Esta seção descreve as principais considerações para continuar sua jornada de modernização.

Esta seção contém os seguintes tópicos:

- Estratégias incrementais para decomposição do banco de dados
- Considerações técnicas para ambientes de banco de dados distribuídos
- Mudanças organizacionais para dar suporte a arquiteturas distribuídas

Estratégias incrementais para decomposição do banco de dados

A decomposição do banco de dados segue uma evolução gradual por meio de três fases distintas. As equipes primeiro agrupam o banco de dados monolítico com um serviço de encapsulamento de banco de dados para controlar o acesso. Em seguida, eles começam a dividir os dados em bancos de dados específicos do serviço, enquanto mantêm o banco de dados principal para as necessidades antigas. Por fim, eles migram completamente a lógica de negócios para fazer a transição para bancos de dados de serviços totalmente independentes.

Ao longo dessa jornada, as equipes devem implementar padrões cuidadosos de sincronização de dados e validar continuamente a consistência entre os serviços. O monitoramento do desempenho se torna crucial para identificar e resolver possíveis problemas com antecedência. À medida que os serviços evoluem de forma independente, seus esquemas devem ser otimizados com base nos padrões de uso reais, e você deve remover estruturas redundantes que se acumularam ao longo do tempo.

Essa abordagem incremental ajuda a minimizar os riscos enquanto mantém a estabilidade do sistema durante todo o processo de transformação.

Considerações técnicas para ambientes de banco de dados distribuídos

Em um ambiente de banco de dados distribuído, o monitoramento do desempenho se torna essencial para identificar e resolver gargalos precocemente. As equipes devem implementar sistemas abrangentes de monitoramento e estratégias de armazenamento em cache para manter os níveis de desempenho. Read/write a divisão pode equilibrar efetivamente as cargas em todo o sistema.

A consistência dos dados exige uma orquestração cuidadosa entre os serviços distribuídos. As equipes devem implementar eventuais padrões de consistência quando apropriado e estabelecer limites claros de propriedade dos dados. O monitoramento robusto promove a integridade dos dados em todos os serviços.

Além disso, a segurança deve evoluir para acomodar a arquitetura distribuída. Cada serviço precisa de controles de segurança refinados, e seus padrões de acesso exigem uma revisão regular. O monitoramento e a auditoria aprimorados se tornam essenciais nesse ambiente distribuído.

Mudanças organizacionais para dar suporte a arquiteturas distribuídas

A estrutura da equipe deve se alinhar aos limites do serviço para definir uma propriedade e responsabilidade claras. As organizações devem estabelecer novos padrões de comunicação e desenvolver capacidades técnicas adicionais dentro das equipes. Essa estrutura deve suportar tanto a manutenção dos serviços existentes quanto sua evolução arquitetônica contínua.

Você deve atualizar seus processos operacionais para lidar com a arquitetura distribuída. As equipes devem modificar os procedimentos de implantação, adaptar os processos de resposta a incidentes e desenvolver as práticas de gerenciamento de mudanças para coordenar vários serviços.

Recursos

Os recursos e ferramentas adicionais a seguir podem ajudar sua organização em sua jornada de decomposição do banco de dados.

AWS Orientação prescritiva

- Migrando Oracle bancos de dados para o Nuvem AWS
- Opções de replataforma para um Oracle DatabaseAWS
- Padrões, arquiteturas e implementações de design de nuvem

AWS postagens no blog

- Migre a lógica de negócios do banco de dados para o aplicativo para acelerar a inovação e a flexibilidade

Serviços da AWS

- AWS Application Migration Service
- AWS Database Migration Service (AWS DMS)
- Migration Evaluator
- AWS Schema Conversion Tool (AWS SCT)
- AWS Transform

Outras ferramentas

- AppEngine (site da Dynatrace)
- Oracle Automatic Workload Repository (site da Oracle)
- CAST Imaging (site da CAST)
- Kiro (site da Kiro)
- pgAdmin (site da pgAdmin)
- pg_stat_statements (site da PostgreSQL)

- SchemaSpy (site da SchemaSpy)
- SQL Developer (site da Oracle)
- SQLWays (site da Ispirer)
- vFunction (site da vFunction)

Outros recursos

- Do monólito aos microsserviços (site) O'Reilly

Histórico do documento

A tabela a seguir descreve alterações significativas feitas neste guia. Se desejar receber notificações sobre futuras atualizações, inscreva-se em um feed RSS.

Alteração	Descrição	Data
Perguntas frequentes sobre mainframe e ferramentas de IA	Adicionamos a pergunta Essas recomendações se aplicam a bancos de dados de mainframe monolíticos? Perguntas frequentes e adicionamos informações adicionais sobre ferramentas de IA que você pode usar durante a decomposição do banco de dados.	14 de outubro de 2025
Publicação inicial	—	30 de setembro de 2025

AWS Glossário de orientação prescritiva

A seguir estão os termos comumente usados em estratégias, guias e padrões fornecidos pela Orientação AWS Prescritiva. Para sugerir entradas, use o link Fornecer feedback no final do glossário.

Números

7 Rs

Sete estratégias comuns de migração para mover aplicações para a nuvem. Essas estratégias baseiam-se nos 5 Rs identificados pela Gartner em 2011 e consistem em:

- Refatorar/rearquitetar: mova uma aplicação e modifique sua arquitetura aproveitando ao máximo os recursos nativos de nuvem para melhorar a agilidade, a performance e a escalabilidade. Isso normalmente envolve a portabilidade do sistema operacional e do banco de dados. Exemplo: migre seu banco de dados Oracle local para a edição compatível com o Amazon Aurora PostgreSQL.
- Redefinir a plataforma (mover e redefinir [mover e redefinir (lift-and-reshape]): mova uma aplicação para a nuvem e introduza algum nível de otimização a fim de aproveitar os recursos da nuvem. Exemplo: Migre seu banco de dados Oracle local para o Amazon Relational Database Service (Amazon RDS) for Oracle no. Nuvem AWS
- Recomprar (drop and shop): mude para um produto diferente, normalmente migrando de uma licença tradicional para um modelo SaaS. Exemplo: migre seu sistema de gerenciamento de relacionamento com o cliente (CRM) para a Salesforce.com.
- Redefinir a hospedagem (mover sem alterações [lift-and-shift])mover uma aplicação para a nuvem sem fazer nenhuma alteração a fim de aproveitar os recursos da nuvem. Exemplo: Migre seu banco de dados Oracle local para o Oracle em uma EC2 instância no. Nuvem AWS
- Realocar (mover o hipervisor sem alterações [hypervisor-level lift-and-shift]): mover a infraestrutura para a nuvem sem comprar novo hardware, reescrever aplicações ou modificar suas operações existentes. Você migra servidores de uma plataforma local para um serviço em nuvem para a mesma plataforma. Exemplo: Migrar um Microsoft Hyper-V aplicativo para o. AWS
- Reter (revisitar): mantenha as aplicações em seu ambiente de origem. Isso pode incluir aplicações que exigem grande refatoração, e você deseja adiar esse trabalho para um

momento posterior, e aplicações antigas que você deseja manter porque não há justificativa comercial para migrá-las.

- Retirar: desative ou remova aplicações que não são mais necessárias em seu ambiente de origem.

A

ABAC

Consulte controle de acesso baseado em atributos.

serviços abstratos

Veja os serviços gerenciados.

ACID

Veja atomicidade, consistência, isolamento, durabilidade.

migração ativa-ativa

Um método de migração de banco de dados no qual os bancos de dados de origem e de destino são mantidos em sincronia (por meio de uma ferramenta de replicação bidirecional ou operações de gravação dupla), e ambos os bancos de dados lidam com transações de aplicações conectadas durante a migração. Esse método oferece suporte à migração em lotes pequenos e controlados, em vez de exigir uma substituição única. É mais flexível, mas exige mais trabalho do que a migração ativa-passiva.

migração ativa-passiva

Um método de migração de banco de dados no qual os bancos de dados de origem e de destino são mantidos em sincronia, mas somente o banco de dados de origem manipula as transações dos aplicativos de conexão enquanto os dados são replicados no banco de dados de destino. O banco de dados de destino não aceita nenhuma transação durante a migração.

função agregada

Uma função SQL que opera em um grupo de linhas e calcula um único valor de retorno para o grupo. Exemplos de funções agregadas incluem `SUM` e. `MAX`

AI

Veja a inteligência artificial.

AIOps

Veja as operações de inteligência artificial.

anonimização

O processo de excluir permanentemente informações pessoais em um conjunto de dados. A anonimização pode ajudar a proteger a privacidade pessoal. Dados anônimos não são mais considerados dados pessoais.

antipadrões

Uma solução frequentemente usada para um problema recorrente em que a solução é contraproducente, ineficaz ou menos eficaz do que uma alternativa.

controle de aplicativos

Uma abordagem de segurança que permite o uso somente de aplicativos aprovados para ajudar a proteger um sistema contra malware.

portfólio de aplicações

Uma coleção de informações detalhadas sobre cada aplicação usada por uma organização, incluindo o custo para criar e manter a aplicação e seu valor comercial. Essas informações são fundamentais para o processo de descoberta e análise de portfólio e ajudam a identificar e priorizar as aplicações a serem migradas, modernizadas e otimizadas.

inteligência artificial (IA)

O campo da ciência da computação que se dedica ao uso de tecnologias de computação para desempenhar funções cognitivas normalmente associadas aos humanos, como aprender, resolver problemas e reconhecer padrões. Para obter mais informações, consulte O que é inteligência artificial?

operações de inteligência artificial (AIOps)

O processo de usar técnicas de machine learning para resolver problemas operacionais, reduzir incidentes operacionais e intervenção humana e aumentar a qualidade do serviço. Para obter mais informações sobre como AIOps é usado na estratégia de AWS migração, consulte o guia de integração de operações.

criptografia assimétrica

Um algoritmo de criptografia que usa um par de chaves, uma chave pública para criptografia e uma chave privada para descriptografia. É possível compartilhar a chave pública porque ela não é usada na descriptografia, mas o acesso à chave privada deve ser altamente restrito.

atomicidade, consistência, isolamento, durabilidade (ACID)

Um conjunto de propriedades de software que garantem a validade dos dados e a confiabilidade operacional de um banco de dados, mesmo no caso de erros, falhas de energia ou outros problemas.

controle de acesso por atributo (ABAC)

A prática de criar permissões minuciosas com base nos atributos do usuário, como departamento, cargo e nome da equipe. Para obter mais informações, consulte ABAC AWS na documentação AWS Identity and Access Management (IAM).

fonte de dados autorizada

Um local onde você armazena a versão principal dos dados, que é considerada a fonte de informações mais confiável. Você pode copiar dados da fonte de dados autorizada para outros locais com o objetivo de processar ou modificar os dados, como anonimizá-los, redigi-los ou pseudonimizá-los.

Zona de disponibilidade

Um local distinto dentro de um Região da AWS que está isolado de falhas em outras zonas de disponibilidade e fornece conectividade de rede barata e de baixa latência a outras zonas de disponibilidade na mesma região.

AWS Estrutura de adoção da nuvem (AWS CAF)

Uma estrutura de diretrizes e melhores práticas AWS para ajudar as organizações a desenvolver um plano eficiente e eficaz para migrar com sucesso para a nuvem. AWS O CAF organiza a orientação em seis áreas de foco chamadas perspectivas: negócios, pessoas, governança, plataforma, segurança e operações. As perspectivas de negócios, pessoas e governança têm como foco habilidades e processos de negócios; as perspectivas de plataforma, segurança e operações concentram-se em habilidades e processos técnicos. Por exemplo, a perspectiva das pessoas tem como alvo as partes interessadas que lidam com recursos humanos (RH), funções de pessoal e gerenciamento de pessoal. Nessa perspectiva, o AWS CAF fornece orientação para desenvolvimento, treinamento e comunicação de pessoas para ajudar a preparar a organização para a adoção bem-sucedida da nuvem. Para obter mais informações, consulte o site da AWS CAF e o whitepaper da AWS CAF.

AWS Estrutura de qualificação da carga de trabalho (AWS WQF)

Uma ferramenta que avalia as cargas de trabalho de migração do banco de dados, recomenda estratégias de migração e fornece estimativas de trabalho. AWS O WQF está incluído com AWS

Schema Conversion Tool ()AWS SCT. Ela analisa esquemas de banco de dados e objetos de código, código de aplicações, dependências e características de performance, além de fornecer relatórios de avaliação.

B

bot ruim

Um bot destinado a perturbar ou causar danos a indivíduos ou organizações.

BCP

Veja o planejamento de continuidade de negócios.

gráfico de comportamento

Uma visualização unificada e interativa do comportamento e das interações de recursos ao longo do tempo. É possível usar um gráfico de comportamento com o Amazon Detective para examinar tentativas de login malsucedidas, chamadas de API suspeitas e ações similares. Para obter mais informações, consulte Dados em um gráfico de comportamento na documentação do Detective.

sistema big-endian

Um sistema que armazena o byte mais significativo antes. Veja também endianness.

classificação binária

Um processo que prevê um resultado binário (uma de duas classes possíveis). Por exemplo, seu modelo de ML pode precisar prever problemas como “Este e-mail é ou não é spam?” ou “Este produto é um livro ou um carro?”

filtro de bloom

Uma estrutura de dados probabilística e eficiente em termos de memória que é usada para testar se um elemento é membro de um conjunto.

blue/green deployment (implantação azul/verde)

Uma estratégia de implantação em que você cria dois ambientes separados, mas idênticos. Você executa a versão atual do aplicativo em um ambiente (azul) e a nova versão do aplicativo no outro ambiente (verde). Essa estratégia ajuda você a reverter rapidamente com o mínimo de impacto.

bot

Um aplicativo de software que executa tarefas automatizadas pela Internet e simula a atividade ou interação humana. Alguns bots são úteis ou benéficos, como rastreadores da Web que indexam informações na Internet. Alguns outros bots, conhecidos como bots ruins, têm como objetivo perturbar ou causar danos a indivíduos ou organizações.

botnet

Redes de bots infectadas por malware e sob o controle de uma única parte, conhecidas como pastor de bots ou operador de bots. As redes de bots são o mecanismo mais conhecido para escalar bots e seu impacto.

ramo

Uma área contida de um repositório de código. A primeira ramificação criada em um repositório é a ramificação principal. Você pode criar uma nova ramificação a partir de uma ramificação existente e, em seguida, desenvolver recursos ou corrigir bugs na nova ramificação. Uma ramificação que você cria para gerar um recurso é comumente chamada de ramificação de recurso. Quando o recurso estiver pronto para lançamento, você mesclará a ramificação do recurso de volta com a ramificação principal. Para obter mais informações, consulte Sobre filiais (GitHub documentação).

acesso em vidro quebrado

Em circunstâncias excepcionais e por meio de um processo aprovado, um meio rápido para um usuário obter acesso a um Conta da AWS que ele normalmente não tem permissão para acessar. Para obter mais informações, consulte o indicador Implementar procedimentos de quebra de vidro na orientação do Well-Architected AWS .

estratégia brownfield

A infraestrutura existente em seu ambiente. Ao adotar uma estratégia brownfield para uma arquitetura de sistema, você desenvolve a arquitetura de acordo com as restrições dos sistemas e da infraestrutura atuais. Se estiver expandindo a infraestrutura existente, poderá combinar as estratégias brownfield e greenfield.

cache do buffer

A área da memória em que os dados acessados com mais frequência são armazenados.

capacidade de negócios

O que uma empresa faz para gerar valor (por exemplo, vendas, atendimento ao cliente ou marketing). As arquiteturas de microsserviços e as decisões de desenvolvimento podem

ser orientadas por recursos de negócios. Para obter mais informações, consulte a seção Organizados de acordo com as capacidades de negócios do whitepaper Executar microsserviços conteinerizados na AWS.

planejamento de continuidade de negócios (BCP)

Um plano que aborda o impacto potencial de um evento disruptivo, como uma migração em grande escala, nas operações e permite que uma empresa retome as operações rapidamente.

C

CAF

Consulte Estrutura de adoção da AWS nuvem.

implantação canária

O lançamento lento e incremental de uma versão para usuários finais. Quando estiver confiante, você implanta a nova versão e substituirá a versão atual em sua totalidade.

CCoE

Veja o Centro de Excelência em Nuvem.

CDC

Veja a captura de dados de alterações.

captura de dados de alterações (CDC)

O processo de rastrear alterações em uma fonte de dados, como uma tabela de banco de dados, e registrar metadados sobre a alteração. É possível usar o CDC para várias finalidades, como auditar ou replicar alterações em um sistema de destino para manter a sincronização.

engenharia do caos

Introduzir intencionalmente falhas ou eventos disruptivos para testar a resiliência de um sistema. Você pode usar AWS Fault Injection Service (AWS FIS) para realizar experimentos que estressam suas AWS cargas de trabalho e avaliar sua resposta.

CI/CD

Veja a integração e a entrega contínuas.

classificação

Um processo de categorização que ajuda a gerar previsões. Os modelos de ML para problemas de classificação predizem um valor discreto. Os valores discretos são sempre diferentes uns dos outros. Por exemplo, um modelo pode precisar avaliar se há ou não um carro em uma imagem.

criptografia no lado do cliente

Criptografia de dados localmente, antes que o alvo os AWS service (Serviço da AWS) receba.

Centro de excelência em nuvem (CCoE)

Uma equipe multidisciplinar que impulsiona os esforços de adoção da nuvem em toda a organização, incluindo o desenvolvimento de práticas recomendas de nuvem, a mobilização de recursos, o estabelecimento de cronogramas de migração e a liderança da organização em transformações em grande escala. Para obter mais informações, consulte as publicações CCo E no Blog de Estratégia Nuvem AWS Empresarial.

computação em nuvem

A tecnologia de nuvem normalmente usada para armazenamento de dados remoto e gerenciamento de dispositivos de IoT. A computação em nuvem geralmente está conectada à tecnologia de computação de ponta.

modelo operacional em nuvem

Em uma organização de TI, o modelo operacional usado para criar, amadurecer e otimizar um ou mais ambientes de nuvem. Para obter mais informações, consulte Criar seu modelo operacional de nuvem.

estágios de adoção da nuvem

As quatro fases pelas quais as organizações normalmente passam quando migram para o Nuvem AWS:

- Projeto: executar alguns projetos relacionados à nuvem para fins de prova de conceito e aprendizado
- Fundação — Fazer investimentos fundamentais para escalar sua adoção da nuvem (por exemplo, criar uma landing zone, definir um CCo E, estabelecer um modelo de operações)
- Migração: migrar aplicações individuais
- Reinvenção: otimizar produtos e serviços e inovar na nuvem

Esses estágios foram definidos por Stephen Orban na postagem do blog The Journey Toward Cloud-First & the Stages of Adoption no blog de estratégia Nuvem AWS empresarial. Para obter

informações sobre como eles se relacionam com a estratégia de AWS migração, consulte o guia de preparação para migração.

CMDB

Consulte o banco de dados de gerenciamento de configuração.

repositório de código

Um local onde o código-fonte e outros ativos, como documentação, amostras e scripts, são armazenados e atualizados por meio de processos de controle de versão. Os repositórios de nuvem comuns incluem GitHub ouBitbucket Cloud. Cada versão do código é chamada de ramificação. Em uma estrutura de microsserviços, cada repositório é dedicado a uma única peça de funcionalidade. Um único pipeline de CI/CD pode usar vários repositórios.

cache frio

Um cache de buffer que está vazio, não está bem preenchido ou contém dados obsoletos ou irrelevantes. Isso afeta a performance porque a instância do banco de dados deve ler da memória principal ou do disco, um processo que é mais lento do que a leitura do cache do buffer.

dados frios

Dados que raramente são acessados e geralmente são históricos. Ao consultar esse tipo de dados, consultas lentas geralmente são aceitáveis. Mover esses dados para níveis ou classes de armazenamento de baixo desempenho e menos caros pode reduzir os custos.

visão computacional (CV)

Um campo da IA que usa aprendizado de máquina para analisar e extrair informações de formatos visuais, como imagens e vídeos digitais. Por exemplo, a Amazon SageMaker AI fornece algoritmos de processamento de imagem para CV.

desvio de configuração

Para uma carga de trabalho, uma alteração de configuração em relação ao estado esperado. Isso pode fazer com que a carga de trabalho se torne incompatível e, normalmente, é gradual e não intencional.

banco de dados de gerenciamento de configuração (CMDB)

Um repositório que armazena e gerencia informações sobre um banco de dados e seu ambiente de TI, incluindo componentes de hardware e software e suas configurações. Normalmente, os dados de um CMDB são usados no estágio de descoberta e análise do portfólio da migração.

pacote de conformidade

Um conjunto de AWS Config regras e ações de remediação que você pode montar para personalizar suas verificações de conformidade e segurança. Você pode implantar um pacote de conformidade como uma entidade única em uma Conta da AWS região ou em uma organização usando um modelo YAML. Para obter mais informações, consulte Pacotes de conformidade na documentação. AWS Config

integração contínua e entrega contínua (CI/CD)

O processo de automatizar os estágios de origem, criação, teste, preparação e produção do processo de lançamento do software. CI/CD é comumente descrito como um pipeline. CI/CD pode ajudá-lo a automatizar processos, melhorar a produtividade, melhorar a qualidade do código e entregar com mais rapidez. Para obter mais informações, consulte Benefícios da entrega contínua. CD também pode significar implantação contínua. Para obter mais informações, consulte Entrega contínua versus implantação contínua.

CV

Veja visão computacional.

D

dados em repouso

Dados estacionários em sua rede, por exemplo, dados que estão em um armazenamento.

classificação de dados

Um processo para identificar e categorizar os dados em sua rede com base em criticalidade e confidencialidade. É um componente crítico de qualquer estratégia de gerenciamento de riscos de segurança cibernética, pois ajuda a determinar os controles adequados de proteção e retenção para os dados. A classificação de dados é um componente do pilar de segurança no AWS Well-Architected Framework. Para obter mais informações, consulte Classificação de dados.

desvio de dados

Uma variação significativa entre os dados de produção e os dados usados para treinar um modelo de ML ou uma alteração significativa nos dados de entrada ao longo do tempo. O desvio de dados pode reduzir a qualidade geral, a precisão e a imparcialidade das previsões do modelo de ML.

dados em trânsito

Dados que estão se movendo ativamente pela sua rede, como entre os recursos da rede.

malha de dados

Uma estrutura arquitetônica que fornece propriedade de dados distribuída e descentralizada com gerenciamento e governança centralizados.

minimização de dados

O princípio de coletar e processar apenas os dados estritamente necessários. Praticar a minimização de dados no Nuvem AWS pode reduzir os riscos de privacidade, os custos e a pegada de carbono de sua análise.

perímetro de dados

Um conjunto de proteções preventivas em seu AWS ambiente que ajudam a garantir que somente identidades confiáveis acessem recursos confiáveis das redes esperadas. Para obter mais informações, consulte Construindo um perímetro de dados em. AWS

pré-processamento de dados

A transformação de dados brutos em um formato que seja facilmente analisado por seu modelo de ML. O pré-processamento de dados pode significar a remoção de determinadas colunas ou linhas e o tratamento de valores ausentes, inconsistentes ou duplicados.

proveniência dos dados

O processo de rastrear a origem e o histórico dos dados ao longo de seu ciclo de vida, por exemplo, como os dados foram gerados, transmitidos e armazenados.

titular dos dados

Um indivíduo cujos dados estão sendo coletados e processados.

data warehouse

Um sistema de gerenciamento de dados que oferece suporte à inteligência comercial, como análises. Os data warehouses geralmente contêm grandes quantidades de dados históricos e geralmente são usados para consultas e análises.

linguagem de definição de dados (DDL)

Instruções ou comandos para criar ou modificar a estrutura de tabelas e objetos em um banco de dados.

linguagem de manipulação de dados (DML)

Instruções ou comandos para modificar (inserir, atualizar e excluir) informações em um banco de dados.

DDL

Consulte a linguagem de definição de banco de dados.

deep ensemble

A combinação de vários modelos de aprendizado profundo para gerar previsões. Os deep ensembles podem ser usados para produzir uma previsão mais precisa ou para estimar a incerteza nas previsões.

Aprendizado profundo

Um subcampo do ML que usa várias camadas de redes neurais artificiais para identificar o mapeamento entre os dados de entrada e as variáveis-alvo de interesse.

defense-in-depth

Uma abordagem de segurança da informação na qual uma série de mecanismos e controles de segurança são cuidadosamente distribuídos por toda a rede de computadores para proteger a confidencialidade, a integridade e a disponibilidade da rede e dos dados nela contidos. Ao adotar essa estratégia AWS, você adiciona vários controles em diferentes camadas da AWS Organizations estrutura para ajudar a proteger os recursos. Por exemplo, uma defense-in-depth abordagem pode combinar autenticação multifatorial, segmentação de rede e criptografia.

administrador delegado

Em AWS Organizations, um serviço compatível pode registrar uma conta de AWS membro para administrar as contas da organização e gerenciar as permissões desse serviço. Essa conta é chamada de administrador delegado para esse serviço Para obter mais informações e uma lista de serviços compatíveis, consulte Serviços que funcionam com o AWS Organizations na documentação do AWS Organizations .

implantação

O processo de criar uma aplicação, novos recursos ou correções de código disponíveis no ambiente de destino. A implantação envolve a implementação de mudanças em uma base de código e, em seguida, a criação e execução dessa base de código nos ambientes da aplicação

ambiente de desenvolvimento

Veja o ambiente.

controle detectivo

Um controle de segurança projetado para detectar, registrar e alertar após a ocorrência de um evento. Esses controles são uma segunda linha de defesa, alertando você sobre eventos de segurança que contornaram os controles preventivos em vigor. Para obter mais informações, consulte Controles detectivos em Como implementar controles de segurança na AWS.

mapeamento do fluxo de valor de desenvolvimento (DVSM)

Um processo usado para identificar e priorizar restrições que afetam negativamente a velocidade e a qualidade em um ciclo de vida de desenvolvimento de software. O DVSM estende o processo de mapeamento do fluxo de valor originalmente projetado para práticas de manufatura enxuta. Ele se concentra nas etapas e equipes necessárias para criar e movimentar valor por meio do processo de desenvolvimento de software.

gêmeo digital

Uma representação virtual de um sistema real, como um prédio, fábrica, equipamento industrial ou linha de produção. Os gêmeos digitais oferecem suporte à manutenção preditiva, ao monitoramento remoto e à otimização da produção.

tabela de dimensões

Em um esquema em estrela, uma tabela menor que contém atributos de dados sobre dados quantitativos em uma tabela de fatos. Os atributos da tabela de dimensões geralmente são campos de texto ou números discretos que se comportam como texto. Esses atributos são comumente usados para restringir consultas, filtrar e rotular conjuntos de resultados.

desastre

Um evento que impede que uma workload ou sistema cumpra seus objetivos de negócios em seu local principal de implantação. Esses eventos podem ser desastres naturais, falhas técnicas ou o resultado de ações humanas, como configuração incorreta não intencional ou ataque de malware.

Recuperação de desastres (RD)

A estratégia e o processo que você usa para minimizar o tempo de inatividade e a perda de dados causados por um desastre. Para obter mais informações, consulte Recuperação de desastres de cargas de trabalho em AWS: Recuperação na nuvem no AWS Well-Architected Framework.

DML

Veja a linguagem de manipulação de banco de dados.

design orientado por domínio

Uma abordagem ao desenvolvimento de um sistema de software complexo conectando seus componentes aos domínios em evolução, ou principais metas de negócios, atendidos por cada componente. Esse conceito foi introduzido por Eric Evans em seu livro, Design orientado por domínio: lidando com a complexidade no coração do software (Boston: Addison-Wesley Professional, 2003). Para obter informações sobre como usar o design orientado por domínio com o padrão strangler fig, consulte Modernizar incrementalmente os serviços web herdados do Microsoft ASP.NET (ASMX) usando contêineres e o Amazon API Gateway.

DR

Veja a recuperação de desastres.

detecção de deriva

Rastreando desvios de uma configuração básica. Por exemplo, você pode usar AWS CloudFormation para detectar desvios nos recursos do sistema ou AWS Control Tower para detectar mudanças em seu landing zone que possam afetar a conformidade com os requisitos de governança.

DVSM

Veja o mapeamento do fluxo de valor do desenvolvimento.

E

EDA

Veja a análise exploratória de dados.

EDI

Veja intercâmbio eletrônico de dados.

computação de borda

A tecnologia que aumenta o poder computacional de dispositivos inteligentes nas bordas de uma rede de IoT. Quando comparada à computação em nuvem, a computação de ponta pode reduzir a latência da comunicação e melhorar o tempo de resposta.

intercâmbio eletrônico de dados (EDI)

A troca automatizada de documentos comerciais entre organizações. Para obter mais informações, consulte O que é intercâmbio eletrônico de dados.

Criptografia

Um processo de computação que transforma dados de texto simples, legíveis por humanos, em texto cifrado.

chave de criptografia

Uma sequência criptográfica de bits aleatórios que é gerada por um algoritmo de criptografia. As chaves podem variar em tamanho, e cada chave foi projetada para ser imprevisível e exclusiva.

endianismo

A ordem na qual os bytes são armazenados na memória do computador. Os sistemas big-endian armazenam o byte mais significativo antes. Os sistemas little-endian armazenam o byte menos significativo antes.

endpoint

Veja o endpoint do serviço.

serviço de endpoint

Um serviço que pode ser hospedado em uma nuvem privada virtual (VPC) para ser compartilhado com outros usuários. Você pode criar um serviço de endpoint com AWS PrivateLink e conceder permissões a outros diretores Contas da AWS ou a AWS Identity and Access Management (IAM). Essas contas ou entidades principais podem se conectar ao serviço de endpoint de maneira privada criando endpoints da VPC de interface. Para obter mais informações, consulte Criar um serviço de endpoint na documentação do Amazon Virtual Private Cloud (Amazon VPC).

planejamento de recursos corporativos (ERP)

Um sistema que automatiza e gerencia os principais processos de negócios (como contabilidade, MES e gerenciamento de projetos) para uma empresa.

criptografia envelopada

O processo de criptografar uma chave de criptografia com outra chave de criptografia. Para obter mais informações, consulte Criptografia de envelope na documentação AWS Key Management Service (AWS KMS).

ambiente

Uma instância de uma aplicação em execução. Estes são tipos comuns de ambientes na computação em nuvem:

- ambiente de desenvolvimento: uma instância de uma aplicação em execução que está disponível somente para a equipe principal responsável pela manutenção da aplicação. Ambientes de desenvolvimento são usados para testar mudanças antes de promovê-las para ambientes superiores. Esse tipo de ambiente às vezes é chamado de ambiente de teste.
- ambientes inferiores: todos os ambientes de desenvolvimento para uma aplicação, como aqueles usados para compilações e testes iniciais.
- ambiente de produção: uma instância de uma aplicação em execução que os usuários finais podem acessar. Em um CI/CD pipeline, o ambiente de produção é o último ambiente de implantação.
- ambientes superiores: todos os ambientes que podem ser acessados por usuários que não sejam a equipe principal de desenvolvimento. Isso pode incluir um ambiente de produção, ambientes de pré-produção e ambientes para testes de aceitação do usuário.

epic

Em metodologias ágeis, categorias funcionais que ajudam a organizar e priorizar seu trabalho. Os epics fornecem uma descrição de alto nível dos requisitos e das tarefas de implementação. Por exemplo, os épicos de segurança AWS da CAF incluem gerenciamento de identidade e acesso, controles de detetive, segurança de infraestrutura, proteção de dados e resposta a incidentes. Para obter mais informações sobre epics na estratégia de migração da AWS , consulte o guia de implementação do programa.

ERP

Veja o planejamento de recursos corporativos.

análise exploratória de dados (EDA)

O processo de analisar um conjunto de dados para entender suas principais características. Você coleta ou agrega dados e, em seguida, realiza investigações iniciais para encontrar padrões, detectar anomalias e verificar suposições. O EDA é realizado por meio do cálculo de estatísticas resumidas e da criação de visualizações de dados.

F

tabela de fatos

A tabela central em um esquema em estrela. Ele armazena dados quantitativos sobre as operações comerciais. Normalmente, uma tabela de fatos contém dois tipos de colunas: aquelas que contêm medidas e aquelas que contêm uma chave externa para uma tabela de dimensões.

falham rapidamente

Uma filosofia que usa testes frequentes e incrementais para reduzir o ciclo de vida do desenvolvimento. É uma parte essencial de uma abordagem ágil.

limite de isolamento de falhas

No Nuvem AWS, um limite, como uma zona de disponibilidade, Região da AWS um plano de controle ou um plano de dados, que limita o efeito de uma falha e ajuda a melhorar a resiliência das cargas de trabalho. Para obter mais informações, consulte Limites de isolamento de AWS falhas.

ramificação de recursos

Veja a filial.

recursos

Os dados de entrada usados para fazer uma previsão. Por exemplo, em um contexto de manufatura, os recursos podem ser imagens capturadas periodicamente na linha de fabricação.

importância do recurso

O quanto um recurso é importante para as previsões de um modelo. Isso geralmente é expresso como uma pontuação numérica que pode ser calculada por meio de várias técnicas, como Shapley Additive Explanations (SHAP) e gradientes integrados. Para obter mais informações, consulte Interpretabilidade do modelo de aprendizado de máquina com AWS.

transformação de recursos

O processo de otimizar dados para o processo de ML, incluindo enriquecer dados com fontes adicionais, escalar valores ou extrair vários conjuntos de informações de um único campo de dados. Isso permite que o modelo de ML se beneficie dos dados. Por exemplo, se a data “2021-05-27 00:15:37” for dividida em “2021”, “maio”, “quinta” e “15”, isso poderá ajudar o algoritmo de aprendizado a aprender padrões diferenciados associados a diferentes componentes de dados.

solicitação rápida

Fornecer a um LLM um pequeno número de exemplos que demonstram a tarefa e o resultado desejado antes de solicitar que ele execute uma tarefa semelhante. Essa técnica é uma aplicação do aprendizado contextual, em que os modelos aprendem com exemplos (fotos) incorporados aos prompts. Solicitações rápidas podem ser eficazes para tarefas que exigem formatação, raciocínio ou conhecimento de domínio específicos. Veja também a solicitação zero-shot.

FGAC

Veja o controle de acesso refinado.

Controle de acesso refinado (FGAC)

O uso de várias condições para permitir ou negar uma solicitação de acesso.

migração flash-cut

Um método de migração de banco de dados que usa replicação contínua de dados por meio da captura de dados alterados para migrar dados no menor tempo possível, em vez de usar uma abordagem em fases. O objetivo é reduzir ao mínimo o tempo de inatividade.

FM

Veja o modelo da fundação.

modelo de fundação (FM)

Uma grande rede neural de aprendizado profundo que vem treinando em grandes conjuntos de dados generalizados e não rotulados. FMs são capazes de realizar uma ampla variedade de tarefas gerais, como entender a linguagem, gerar texto e imagens e conversar em linguagem natural. Para obter mais informações, consulte O que são modelos básicos.

G

IA generativa

Um subconjunto de modelos de IA que foram treinados em grandes quantidades de dados e que podem usar uma simples solicitação de texto para criar novos conteúdos e artefatos, como imagens, vídeos, texto e áudio. Para obter mais informações, consulte O que é IA generativa.

bloqueio geográfico

Veja as restrições geográficas.

restrições geográficas (bloqueio geográfico)

Na Amazon CloudFront, uma opção para impedir que usuários em países específicos acessem distribuições de conteúdo. É possível usar uma lista de permissões ou uma lista de bloqueios para especificar países aprovados e banidos. Para obter mais informações, consulte Restringir a distribuição geográfica do seu conteúdo na CloudFront documentação.

Fluxo de trabalho do GitFlow

Uma abordagem na qual ambientes inferiores e superiores usam ramificações diferentes em um repositório de código-fonte. O fluxo de trabalho do Gitflow é considerado legado, e o fluxo de trabalho baseado em troncos é a abordagem moderna e preferida.

imagem dourada

Um instantâneo de um sistema ou software usado como modelo para implantar novas instâncias desse sistema ou software. Por exemplo, na manufatura, uma imagem dourada pode ser usada para provisionar software em vários dispositivos e ajudar a melhorar a velocidade, a escalabilidade e a produtividade nas operações de fabricação de dispositivos.

estratégia greenfield

A ausência de infraestrutura existente em um novo ambiente. Ao adotar uma estratégia greenfield para uma arquitetura de sistema, é possível selecionar todas as novas tecnologias sem a restrição da compatibilidade com a infraestrutura existente, também conhecida como brownfield. Se estiver expandindo a infraestrutura existente, poderá combinar as estratégias brownfield e greenfield.

barreira de proteção

Uma regra de alto nível que ajuda a governar recursos, políticas e conformidade em todas as unidades organizacionais ()OUs. Barreiras de proteção preventivas impõem políticas para garantir o alinhamento a padrões de conformidade. Elas são implementadas usando políticas de controle de serviço e limites de permissões do IAM. Barreiras de proteção detectivas detectam violações de políticas e problemas de conformidade e geram alertas para remediação. Eles são implementados usando AWS Config, AWS Security Hub CSPM, Amazon GuardDuty AWS Trusted Advisor, Amazon Inspector e verificações personalizadas AWS Lambda .

H

HA

Veja a alta disponibilidade.

migração heterogênea de bancos de dados

Migrar seu banco de dados de origem para um banco de dados de destino que usa um mecanismo de banco de dados diferente (por exemplo, Oracle para Amazon Aurora). A migração heterogênea geralmente faz parte de um esforço de redefinição da arquitetura, e converter

o esquema pode ser uma tarefa complexa. O AWS fornece o AWS SCT para ajudar nas conversões de esquemas.

alta disponibilidade (HA)

A capacidade de uma workload operar continuamente, sem intervenção, em caso de desafios ou desastres. Os sistemas AH são projetados para realizar o failover automático, oferecer consistentemente desempenho de alta qualidade e lidar com diferentes cargas e falhas com impacto mínimo no desempenho.

modernização de historiador

Uma abordagem usada para modernizar e atualizar os sistemas de tecnologia operacional (OT) para melhor atender às necessidades do setor de manufatura. Um historiador é um tipo de banco de dados usado para coletar e armazenar dados de várias fontes em uma fábrica.

dados de retenção

Uma parte dos dados históricos rotulados que são retidos de um conjunto de dados usado para treinar um modelo de aprendizado de máquina. Você pode usar dados de retenção para avaliar o desempenho do modelo comparando as previsões do modelo com os dados de retenção.

migração homogênea de bancos de dados

Migrar seu banco de dados de origem para um banco de dados de destino que compartilha o mesmo mecanismo de banco de dados (por exemplo, Microsoft SQL Server para Amazon RDS para SQL Server). A migração homogênea geralmente faz parte de um esforço de redefinição da hospedagem ou da plataforma. É possível usar utilitários de banco de dados nativos para migrar o esquema.

dados quentes

Dados acessados com frequência, como dados em tempo real ou dados translacionais recentes. Esses dados normalmente exigem uma camada ou classe de armazenamento de alto desempenho para fornecer respostas rápidas às consultas.

hotfix

Uma correção urgente para um problema crítico em um ambiente de produção. Devido à sua urgência, um hotfix geralmente é feito fora do fluxo de trabalho normal de DevOps lançamento.

período de hipercuidados

Imediatamente após a substituição, o período em que uma equipe de migração gerencia e monitora as aplicações migradas na nuvem para resolver quaisquer problemas. Normalmente,

a duração desse período é de 1 a 4 dias. No final do período de hipercuidados, a equipe de migração normalmente transfere a responsabilidade pelas aplicações para a equipe de operações de nuvem.

eu

IaC

Veja a infraestrutura como código.

Política baseada em identidade

Uma política anexada a um ou mais diretores do IAM que define suas permissões no Nuvem AWS ambiente.

aplicação ociosa

Uma aplicação que tem um uso médio de CPU e memória entre 5 e 20% em um período de 90 dias. Em um projeto de migração, é comum retirar essas aplicações ou retê-las on-premises.

IIoT

Veja a Internet das Coisas industrial.

infraestrutura imutável

Um modelo que implanta uma nova infraestrutura para cargas de trabalho de produção em vez de atualizar, corrigir ou modificar a infraestrutura existente. Infraestruturas imutáveis são inerentemente mais consistentes, confiáveis e previsíveis do que infraestruturas mutáveis. Para obter mais informações, consulte as melhores práticas de implantação usando infraestrutura imutável no Well-Architected AWS Framework.

VPC de entrada (admissão)

Em uma arquitetura de AWS várias contas, uma VPC que aceita, inspeciona e roteia conexões de rede de fora de um aplicativo. A Arquitetura de Referência de AWS Segurança recomenda configurar sua conta de rede com entrada, saída e inspeção VPCs para proteger a interface bidirecional entre seu aplicativo e a Internet em geral.

migração incremental

Uma estratégia de substituição na qual você migra a aplicação em pequenas partes, em vez de realizar uma única substituição completa. Por exemplo, é possível mover inicialmente

apenas alguns microsserviços ou usuários para o novo sistema. Depois de verificar se tudo está funcionando corretamente, mova os microsserviços ou usuários adicionais de forma incremental até poder descomissionar seu sistema herdado. Essa estratégia reduz os riscos associados a migrações de grande porte.

Indústria 4.0

Um termo que foi introduzido por Klaus Schwab em 2016 para se referir à modernização dos processos de fabricação por meio de avanços em conectividade, dados em tempo real, automação, análise e IA/ML.

infraestrutura

Todos os recursos e ativos contidos no ambiente de uma aplicação.

Infraestrutura como código (IaC)

O processo de provisionamento e gerenciamento da infraestrutura de uma aplicação por meio de um conjunto de arquivos de configuração. A IaC foi projetada para ajudar você a centralizar o gerenciamento da infraestrutura, padronizar recursos e escalar rapidamente para que novos ambientes sejam reproduzíveis, confiáveis e consistentes.

Internet industrial das coisas (IIoT)

O uso de sensores e dispositivos conectados à Internet nos setores industriais, como manufatura, energia, automotivo, saúde, ciências biológicas e agricultura. Para obter mais informações, consulte Criando uma estratégia de transformação digital industrial da Internet das Coisas (IIoT).

VPC de inspeção

Em uma arquitetura de AWS várias contas, uma VPC centralizada que gerencia as inspeções do tráfego de rede entre VPCs (na mesma ou em diferentes Regiões da AWS) a Internet e as redes locais. A Arquitetura de Referência de AWS Segurança recomenda configurar sua conta de rede com entrada, saída e inspeção VPCs para proteger a interface bidirecional entre seu aplicativo e a Internet em geral.

Internet das coisas (IoT)

A rede de objetos físicos conectados com sensores ou processadores incorporados que se comunicam com outros dispositivos e sistemas pela Internet ou por uma rede de comunicação local. Para obter mais informações, consulte O que é IoT?

interpretabilidade

Uma característica de um modelo de machine learning que descreve o grau em que um ser humano pode entender como as previsões do modelo dependem de suas entradas. Para obter mais informações, consulte Interpretabilidade do modelo de aprendizado de máquina com AWS.

IoT

Consulte Internet das Coisas.

Biblioteca de informações de TI (ITIL)

Um conjunto de práticas recomendadas para fornecer serviços de TI e alinhar esses serviços a requisitos de negócios. A ITIL fornece a base para o ITSM.

Gerenciamento de serviços de TI (ITSM)

Atividades associadas a design, implementação, gerenciamento e suporte de serviços de TI para uma organização. Para obter informações sobre a integração de operações em nuvem com ferramentas de ITSM, consulte o guia de integração de operações.

ITIL

Consulte a biblioteca de informações de TI.

ITSM

Veja o gerenciamento de serviços de TI.

L

controle de acesso baseado em etiqueta (LBAC)

Uma implementação do controle de acesso obrigatório (MAC) em que os usuários e os dados em si recebem explicitamente um valor de etiqueta de segurança. A interseção entre a etiqueta de segurança do usuário e a etiqueta de segurança dos dados determina quais linhas e colunas podem ser vistas pelo usuário.

zona de pouso

Uma landing zone é um AWS ambiente bem arquitetado, com várias contas, escalável e seguro. Um ponto a partir do qual suas organizações podem iniciar e implantar rapidamente workloads e aplicações com confiança em seu ambiente de segurança e infraestrutura. Para obter mais

informações sobre zonas de pouso, consulte Configurar um ambiente da AWS com várias contas seguro e escalável.

modelo de linguagem grande (LLM)

Um modelo de IA de aprendizado profundo pré-treinado em uma grande quantidade de dados. Um LLM pode realizar várias tarefas, como responder perguntas, resumir documentos, traduzir texto para outros idiomas e completar frases. Para obter mais informações, consulte O que são LLMs.

migração de grande porte

Uma migração de 300 servidores ou mais.

LBAC

Veja controle de acesso baseado em etiquetas.

privilégio mínimo

A prática recomendada de segurança de conceder as permissões mínimas necessárias para executar uma tarefa. Para obter mais informações, consulte Aplicar permissões de privilégios mínimos na documentação do IAM.

mover sem alterações (lift-and-shift)

Veja 7 Rs.

sistema little-endian

Um sistema que armazena o byte menos significativo antes. Veja também endianness.

LLM

Veja um modelo de linguagem grande.

ambientes inferiores

Veja o ambiente.

M

machine learning (ML)

Um tipo de inteligência artificial que usa algoritmos e técnicas para reconhecimento e aprendizado de padrões. O ML analisa e aprende com dados gravados, por exemplo, dados da

Internet das Coisas (IoT), para gerar um modelo estatístico baseado em padrões. Para obter mais informações, consulte Machine learning.

ramificação principal

Veja a filial.

malware

Software projetado para comprometer a segurança ou a privacidade do computador. O malware pode interromper os sistemas do computador, vazar informações confidenciais ou obter acesso não autorizado. Exemplos de malware incluem vírus, worms, ransomware, cavalos de Tróia, spyware e keyloggers.

serviços gerenciados

Serviços da AWS para o qual AWS opera a camada de infraestrutura, o sistema operacional e as plataformas, e você acessa os endpoints para armazenar e recuperar dados. O Amazon Simple Storage Service (Amazon S3) e o Amazon DynamoDB são exemplos de serviços gerenciados. Eles também são conhecidos como serviços abstratos.

sistema de execução de manufatura (MES)

Um sistema de software para rastrear, monitorar, documentar e controlar processos de produção que convertem matérias-primas em produtos acabados no chão de fábrica.

MAP

Consulte Migration Acceleration Program.

mecanismo

Um processo completo no qual você cria uma ferramenta, impulsiona a adoção da ferramenta e, em seguida, inspeciona os resultados para fazer ajustes. Um mecanismo é um ciclo que se reforça e se aprimora à medida que opera. Para obter mais informações, consulte Construindo mecanismos no AWS Well-Architected Framework.

conta de membro

Todos, Contas da AWS exceto a conta de gerenciamento, que fazem parte de uma organização em AWS Organizations. Uma conta só pode ser membro de uma organização de cada vez.

MES

Veja o sistema de execução de manufatura.

Transporte de telemetria de enfileiramento de mensagens (MQTT)

Um protocolo de comunicação leve machine-to-machine (M2M), baseado no padrão de publicação/assinatura, para dispositivos de IoT com recursos limitados.

microsserviço

Um serviço pequeno e independente que se comunica de forma bem definida APIs e normalmente é de propriedade de equipes pequenas e independentes. Por exemplo, um sistema de seguradora pode incluir microsserviços que mapeiam as capacidades comerciais, como vendas ou marketing, ou subdomínios, como compras, reclamações ou análises. Os benefícios dos microsserviços incluem agilidade, escalabilidade flexível, fácil implantação, código reutilizável e resiliência. Para obter mais informações, consulte Integração de microsserviços usando serviços sem AWS servidor.

arquitetura de microsserviços

Uma abordagem à criação de aplicações com componentes independentes que executam cada processo de aplicação como um microsserviço. Esses microsserviços se comunicam por meio de uma interface bem definida usando leveza. APIs Cada microsserviço nessa arquitetura pode ser atualizado, implantado e escalado para atender à demanda por funções específicas de uma aplicação. Para obter mais informações, consulte Implementação de microsserviços em. AWS

Programa de Aceleração da Migração (MAP)

Um AWS programa que fornece suporte de consultoria, treinamento e serviços para ajudar as organizações a criar uma base operacional sólida para migrar para a nuvem e ajudar a compensar o custo inicial das migrações. O MAP inclui uma metodologia de migração para executar migrações legadas de forma metódica e um conjunto de ferramentas para automatizar e acelerar cenários comuns de migração.

migração em escala

O processo de mover a maior parte do portfólio de aplicações para a nuvem em ondas, com mais aplicações sendo movidas em um ritmo mais rápido a cada onda. Essa fase usa as práticas recomendadas e lições aprendidas nas fases anteriores para implementar uma fábrica de migração de equipes, ferramentas e processos para agilizar a migração de workloads por meio de automação e entrega ágeis. Esta é a terceira fase da estratégia de migração para a AWS.

fábrica de migração

Equipes multifuncionais que simplificam a migração de workloads por meio de abordagens automatizadas e ágeis. As equipes da fábrica de migração geralmente incluem operações,

analistas e proprietários de negócios, engenheiros de migração, desenvolvedores e DevOps profissionais que trabalham em sprints. Entre 20 e 50% de um portfólio de aplicações corporativas consiste em padrões repetidos que podem ser otimizados por meio de uma abordagem de fábrica. Para obter mais informações, consulte discussão sobre fábricas de migração e o guia do Cloud Migration Factory neste conjunto de conteúdo.

metadados de migração

As informações sobre a aplicação e o servidor necessárias para concluir a migração. Cada padrão de migração exige um conjunto de metadados de migração diferente. Exemplos de metadados de migração incluem a sub-rede, o grupo de segurança e AWS a conta de destino.

padrão de migração

Uma tarefa de migração repetível que detalha a estratégia de migração, o destino da migração e a aplicação ou o serviço de migração usado. Exemplo: rehospede a migração para a Amazon EC2 com o AWS Application Migration Service.

Avaliação de Portfólio para Migração (MPA)

Uma ferramenta on-line que fornece informações para validar o caso de negócios para migrar para o. Nuvem AWS O MPA fornece avaliação detalhada do portfólio (dimensionamento correto do servidor, preços, comparações de TCO, análise de custos de migração), bem como planejamento de migração (análise e coleta de dados de aplicações, agrupamento de aplicações, priorização de migração e planejamento de ondas). A ferramenta MPA (requer login) está disponível gratuitamente para todos os AWS consultores e consultores parceiros da APN.

Avaliação de Preparação para Migração (MRA)

O processo de obter insights sobre o status de prontidão de uma organização para a nuvem, identificar pontos fortes e fracos e criar um plano de ação para fechar as lacunas identificadas, usando o CAF. AWS Para mais informações, consulte o guia de preparação para migração. A MRA é a primeira fase da estratégia de migração para a AWS.

estratégia de migração

A abordagem usada para migrar uma carga de trabalho para o. Nuvem AWS Para obter mais informações, consulte a entrada de 7 Rs neste glossário e consulte Mobilize sua organização para acelerar migrações em grande escala.

ML

Veja o aprendizado de máquina.

modernização

Transformar uma aplicação desatualizada (herdada ou monolítica) e sua infraestrutura em um sistema ágil, elástico e altamente disponível na nuvem para reduzir custos, ganhar eficiência e aproveitar as inovações. Para obter mais informações, consulte Estratégia para modernizar aplicativos no Nuvem AWS.

avaliação de preparação para modernização

Uma avaliação que ajuda a determinar a preparação para modernização das aplicações de uma organização. Ela identifica benefícios, riscos e dependências e determina o quão bem a organização pode acomodar o estado futuro dessas aplicações. O resultado da avaliação é um esquema da arquitetura de destino, um roteiro que detalha as fases de desenvolvimento e os marcos do processo de modernização e um plano de ação para abordar as lacunas identificadas. Para obter mais informações, consulte Avaliação da prontidão para modernização de aplicativos no. Nuvem AWS

aplicações monolíticas (monólitos)

Aplicações que são executadas como um único serviço com processos fortemente acoplados. As aplicações monolíticas apresentam várias desvantagens. Se um recurso da aplicação apresentar um aumento na demanda, toda a arquitetura deverá ser escalada. Adicionar ou melhorar os recursos de uma aplicação monolítica também se torna mais complexo quando a base de código cresce. Para resolver esses problemas, é possível criar uma arquitetura de microsserviços. Para obter mais informações, consulte Decompor monólitos em microsserviços.

MPA

Consulte Avaliação do portfólio de migração.

MQTT

Consulte Transporte de telemetria de enfileiramento de mensagens.

classificação multiclasse

Um processo que ajuda a gerar previsões para várias classes (prevendo um ou mais de dois resultados). Por exemplo, um modelo de ML pode perguntar “Este produto é um livro, um carro ou um telefone?” ou “Qual categoria de produtos é mais interessante para este cliente?”

infraestrutura mutável

Um modelo que atualiza e modifica a infraestrutura existente para cargas de trabalho de produção. Para melhorar a consistência, confiabilidade e previsibilidade, o AWS Well-Architected Framework recomenda o uso de infraestrutura imutável como uma prática recomendada.

O

OAC

Veja o controle de acesso de origem.

CARVALHO

Veja a identidade de acesso de origem.

OCM

Veja o gerenciamento de mudanças organizacionais.

migração offline

Um método de migração no qual a workload de origem é desativada durante o processo de migração. Esse método envolve tempo de inatividade prolongado e geralmente é usado para workloads pequenas e não críticas.

OI

Veja a integração de operações.

OLA

Veja o contrato em nível operacional.

migração online

Um método de migração no qual a workload de origem é copiada para o sistema de destino sem ser colocada offline. As aplicações conectadas à workload podem continuar funcionando durante a migração. Esse método envolve um tempo de inatividade nulo ou mínimo e normalmente é usado para workloads essenciais para a produção.

OPC-UA

Consulte Comunicação de processo aberto — Arquitetura unificada.

Comunicação de processo aberto — Arquitetura unificada (OPC-UA)

Um protocolo de comunicação machine-to-machine (M2M) para automação industrial. O OPC-UA fornece um padrão de interoperabilidade com esquemas de criptografia, autenticação e autorização de dados.

acordo de nível operacional (OLA)

Um acordo que esclarece o que os grupos funcionais de TI prometem oferecer uns aos outros para apoiar um acordo de serviço (SLA).

análise de prontidão operacional (ORR)

Uma lista de verificação de perguntas e melhores práticas associadas que ajudam você a entender, avaliar, prevenir ou reduzir o escopo de incidentes e possíveis falhas. Para obter mais informações, consulte Operational Readiness Reviews (ORR) no Well-Architected AWS Framework.

tecnologia operacional (OT)

Sistemas de hardware e software que funcionam com o ambiente físico para controlar operações, equipamentos e infraestrutura industriais. Na manufatura, a integração dos sistemas OT e de tecnologia da informação (TI) é o foco principal das transformações da Indústria 4.0.

integração de operações (OI)

O processo de modernização das operações na nuvem, que envolve planejamento de preparação, automação e integração. Para obter mais informações, consulte o guia de integração de operações.

trilha organizacional

Uma trilha criada por ela AWS CloudTrail registra todos os eventos de todos Contas da AWS em uma organização em AWS Organizations. Essa trilha é criada em cada Conta da AWS que faz parte da organização e monitora a atividade em cada conta. Para obter mais informações, consulte Criação de uma trilha para uma organização na CloudTrail documentação.

gerenciamento de alterações organizacionais (OCM)

Uma estrutura para gerenciar grandes transformações de negócios disruptivas de uma perspectiva de pessoas, cultura e liderança. O OCM ajuda as organizações a se prepararem e fazerem a transição para novos sistemas e estratégias, acelerando a adoção de alterações, abordando questões de transição e promovendo mudanças culturais e organizacionais. Na estratégia de AWS migração, essa estrutura é chamada de aceleração de pessoas, devido à velocidade de mudança exigida nos projetos de adoção da nuvem. Para obter mais informações, consulte o guia do OCM.

controle de acesso de origem (OAC)

Em CloudFront, uma opção aprimorada para restringir o acesso para proteger seu conteúdo do Amazon Simple Storage Service (Amazon S3). O OAC oferece suporte a todos os buckets

S3 Regiões da AWS, criptografia do lado do servidor com AWS KMS (SSE-KMS) e solicitações dinâmicas ao bucket S3. `PUT` `DELETE`

Identidade do acesso de origem (OAI)

Em CloudFront, uma opção para restringir o acesso para proteger seu conteúdo do Amazon S3. Quando você usa o OAI, CloudFront cria um principal com o qual o Amazon S3 pode se autenticar. Os diretores autenticados podem acessar o conteúdo em um bucket do S3 somente por meio de uma distribuição específica. CloudFront Veja também OAC, que fornece um controle de acesso mais granular e aprimorado.

ORR

Veja a análise de prontidão operacional.

OT

Veja a tecnologia operacional.

VPC de saída (egresso)

Em uma arquitetura de AWS várias contas, uma VPC que gerencia conexões de rede que são iniciadas de dentro de um aplicativo. A Arquitetura de Referência de AWS Segurança recomenda configurar sua conta de rede com entrada, saída e inspeção VPCs para proteger a interface bidirecional entre seu aplicativo e a Internet em geral.

P

limite de permissões

Uma política de gerenciamento do IAM anexada a entidades principais do IAM para definir as permissões máximas que o usuário ou perfil podem ter. Para obter mais informações, consulte Limites de permissões na documentação do IAM.

Informações de identificação pessoal (PII)

Informações que, quando visualizadas diretamente ou combinadas com outros dados relacionados, podem ser usadas para inferir razoavelmente a identidade de um indivíduo. Exemplos de PII incluem nomes, endereços e informações de contato.

PII

Veja as informações de identificação pessoal.

manual

Um conjunto de etapas predefinidas que capturam o trabalho associado às migrações, como a entrega das principais funções operacionais na nuvem. Um manual pode assumir a forma de scripts, runbooks automatizados ou um resumo dos processos ou etapas necessários para operar seu ambiente modernizado.

PLC

Consulte controlador lógico programável.

AMEIXA

Veja o gerenciamento do ciclo de vida do produto.

política

Um objeto que pode definir permissões (consulte a política baseada em identidade), especificar as condições de acesso (consulte a política baseada em recursos) ou definir as permissões máximas para todas as contas em uma organização em AWS Organizations (consulte a política de controle de serviços).

persistência poliglota

Escolher de forma independente a tecnologia de armazenamento de dados de um microsserviço com base em padrões de acesso a dados e outros requisitos. Se seus microsserviços tiverem a mesma tecnologia de armazenamento de dados, eles poderão enfrentar desafios de implementação ou apresentar baixa performance. Os microsserviços serão implementados com mais facilidade e alcançarão performance e escalabilidade melhores se usarem o armazenamento de dados mais bem adaptado às suas necessidades. Para obter mais informações, consulte Habilitar a persistência de dados em microsserviços.

avaliação do portfólio

Um processo de descobrir, analisar e priorizar o portfólio de aplicações para planejar a migração. Para obter mais informações, consulte Avaliar a preparação para a migração.

predicado

Uma condição de consulta que retorna `true` ou `false`, normalmente localizada em uma `WHERE` cláusula.

pressão de predicados

Uma técnica de otimização de consulta de banco de dados que filtra os dados na consulta antes da transferência. Isso reduz a quantidade de dados que devem ser recuperados e processados do banco de dados relacional e melhora o desempenho das consultas.

controle preventivo

Um controle de segurança projetado para evitar que um evento ocorra. Esses controles são a primeira linha de defesa para ajudar a evitar acesso não autorizado ou alterações indesejadas em sua rede. Para obter mais informações, consulte Controles preventivos em Como implementar controles de segurança na AWS.

principal (entidade principal)

Uma entidade AWS que pode realizar ações e acessar recursos. Essa entidade geralmente é um usuário raiz para um Conta da AWS, uma função do IAM ou um usuário. Para obter mais informações, consulte Entidade principal em Termos e conceitos de perfis na documentação do IAM.

privacidade por design

Uma abordagem de engenharia de sistema que leva em consideração a privacidade em todo o processo de desenvolvimento.

zonas hospedadas privadas

Um contêiner que contém informações sobre como você deseja que o Amazon Route 53 responda às consultas de DNS para um domínio e seus subdomínios em um ou mais. VPCs Para obter mais informações, consulte Como trabalhar com zonas hospedadas privadas na documentação do Route 53.

controle proativo

Um controle de segurança projetado para impedir a implantação de recursos não compatíveis. Esses controles examinam os recursos antes de serem provisionados. Se o recurso não estiver em conformidade com o controle, ele não será provisionado. Para obter mais informações, consulte o guia de referência de controles na AWS Control Tower documentação e consulte Controles proativos em Implementação de controles de segurança em AWS.

gerenciamento do ciclo de vida do produto (PLM)

O gerenciamento de dados e processos de um produto em todo o seu ciclo de vida, desde o design, desenvolvimento e lançamento, passando pelo crescimento e maturidade, até o declínio e a remoção.

ambiente de produção

Veja o ambiente.

controlador lógico programável (PLC)

Na fabricação, um computador altamente confiável e adaptável que monitora as máquinas e automatiza os processos de fabricação.

encadeamento imediato

Usando a saída de um prompt do LLM como entrada para o próximo prompt para gerar respostas melhores. Essa técnica é usada para dividir uma tarefa complexa em subtarefas ou para refinar ou expandir iterativamente uma resposta preliminar. Isso ajuda a melhorar a precisão e a relevância das respostas de um modelo e permite resultados mais granulares e personalizados.

pseudonimização

O processo de substituir identificadores pessoais em um conjunto de dados por valores de espaço reservado. A pseudonimização pode ajudar a proteger a privacidade pessoal. Os dados pseudonimizados ainda são considerados dados pessoais.

publish/subscribe (pub/sub)

Um padrão que permite comunicações assíncronas entre microsserviços para melhorar a escalabilidade e a capacidade de resposta. Por exemplo, em um MES baseado em microsserviços, um microsserviço pode publicar mensagens de eventos em um canal no qual outros microsserviços possam se inscrever. O sistema pode adicionar novos microsserviços sem alterar o serviço de publicação.

Q

plano de consulta

Uma série de etapas, como instruções, usadas para acessar os dados em um sistema de banco de dados relacional SQL.

regressão de planos de consultas

Quando um otimizador de serviço de banco de dados escolhe um plano menos adequado do que escolhia antes de uma determinada alteração no ambiente de banco de dados ocorrer. Isso pode ser causado por alterações em estatísticas, restrições, configurações do ambiente, associações de parâmetros de consulta e atualizações do mecanismo de banco de dados.

R

Matriz RACI

Veja responsável, responsável, consultado, informado (RACI).

RAG

Consulte Geração Aumentada de Recuperação.

ransomware

Um software mal-intencionado desenvolvido para bloquear o acesso a um sistema ou dados de computador até que um pagamento seja feito.

Matriz RASCI

Veja responsável, responsável, consultado, informado (RACI).

RCAC

Veja o controle de acesso por linha e coluna.

réplica de leitura

Uma cópia de um banco de dados usada somente para leitura. É possível encaminhar consultas para a réplica de leitura e reduzir a carga no banco de dados principal.

rearquiteta

Veja 7 Rs.

objetivo de ponto de recuperação (RPO).

O máximo período de tempo aceitável desde o último ponto de recuperação de dados. Isso determina o que é considerado uma perda aceitável de dados entre o último ponto de recuperação e a interrupção do serviço.

objetivo de tempo de recuperação (RTO)

O máximo atraso aceitável entre a interrupção e a restauração do serviço.

refatorar

Veja 7 Rs.

Região

Uma coleção de AWS recursos em uma área geográfica. Cada um Região da AWS é isolado e independente dos outros para fornecer tolerância a falhas, estabilidade e resiliência. Para obter mais informações, consulte Especificar o que Regiões da AWS sua conta pode usar.

regressão

Uma técnica de ML que prevê um valor numérico. Por exemplo, para resolver o problema de "Por qual preço esta casa será vendida?" um modelo de ML pode usar um modelo de regressão linear para prever o preço de venda de uma casa com base em fatos conhecidos sobre a casa (por exemplo, a metragem quadrada).

redefinir a hospedagem

Veja 7 Rs.

versão

Em um processo de implantação, o ato de promover mudanças em um ambiente de produção.

realocar

Veja 7 Rs.

redefinir a plataforma

Veja 7 Rs.

recomprar

Veja 7 Rs.

resiliência

A capacidade de um aplicativo de resistir ou se recuperar de interrupções. Alta disponibilidade e recuperação de desastres são considerações comuns ao planejar a resiliência no. Nuvem AWS Para obter mais informações, consulte Nuvem AWS Resiliência.

política baseada em recurso

Uma política associada a um recurso, como um bucket do Amazon S3, um endpoint ou uma chave de criptografia. Esse tipo de política especifica quais entidades principais têm acesso permitido, ações válidas e quaisquer outras condições que devem ser atendidas.

matriz responsável, accountable, consultada, informada (RACI)

Uma matriz que define as funções e responsabilidades de todas as partes envolvidas nas atividades de migração e nas operações de nuvem. O nome da matriz é derivado dos tipos de responsabilidade definidos na matriz: responsável (R), responsabilizável (A), consultado (C) e informado (I). O tipo de suporte (S) é opcional. Se você incluir suporte, a matriz será chamada de matriz RASCI e, se excluir, será chamada de matriz RACI.

controle responsivo

Um controle de segurança desenvolvido para conduzir a remediação de eventos adversos ou desvios em relação à linha de base de segurança. Para obter mais informações, consulte Controles responsivos em Como implementar controles de segurança na AWS.

reter

Veja 7 Rs.

aposentar-se

Veja 7 Rs.

Geração Aumentada de Recuperação (RAG)

Uma tecnologia de IA generativa na qual um LLM faz referência a uma fonte de dados autorizada que está fora de suas fontes de dados de treinamento antes de gerar uma resposta. Por exemplo, um modelo RAG pode realizar uma pesquisa semântica na base de conhecimento ou nos dados personalizados de uma organização. Para obter mais informações, consulte O que é RAG.

alternância

O processo de atualizar periodicamente um segredo para dificultar o acesso das credenciais por um invasor.

controle de acesso por linha e coluna (RCAC)

O uso de expressões SQL básicas e flexíveis que tenham regras de acesso definidas. O RCAC consiste em permissões de linha e máscaras de coluna.

RPO

Veja o objetivo do ponto de recuperação.

RTO

Veja o objetivo do tempo de recuperação.

runbook

Um conjunto de procedimentos manuais ou automatizados necessários para realizar uma tarefa específica. Eles são normalmente criados para agilizar operações ou procedimentos repetitivos com altas taxas de erro.

S

SAML 2.0

Um padrão aberto que muitos provedores de identidade (IdPs) usam. Esse recurso permite o login único federado (SSO), para que os usuários possam fazer login Console de gerenciamento da AWS ou chamar as operações da AWS API sem que você precise criar um usuário no IAM para todos em sua organização. Para obter mais informações sobre a federação baseada em SAML 2.0, consulte Sobre a federação baseada em SAML 2.0 na documentação do IAM.

SCADA

Veja controle de supervisão e aquisição de dados.

SCP

Veja a política de controle de serviços.

secret

Em AWS Secrets Manager, informações confidenciais ou restritas, como uma senha ou credenciais de usuário, que você armazena de forma criptografada. Ele consiste no valor secreto e em seus metadados. O valor secreto pode ser binário, uma única string ou várias strings. Para obter mais informações, consulte O que há em um segredo do Secrets Manager? na documentação do Secrets Manager.

segurança por design

Uma abordagem de engenharia de sistema que leva em consideração a segurança em todo o processo de desenvolvimento.

controle de segurança

Uma barreira de proteção técnica ou administrativa que impede, detecta ou reduz a capacidade de uma ameaça explorar uma vulnerabilidade de segurança. Existem quatro tipos principais de controles de segurança: preventivos, detectivos, responsivos e proativos.

fortalecimento da segurança

O processo de reduzir a superfície de ataque para torná-la mais resistente a ataques. Isso pode incluir ações como remover recursos que não são mais necessários, implementar a prática recomendada de segurança de conceder privilégios mínimos ou desativar recursos desnecessários em arquivos de configuração.

sistema de gerenciamento de eventos e informações de segurança (SIEM)

Ferramentas e serviços que combinam sistemas de gerenciamento de informações de segurança (SIM) e gerenciamento de eventos de segurança (SEM). Um sistema SIEM coleta, monitora e analisa dados de servidores, redes, dispositivos e outras fontes para detectar ameaças e violações de segurança e gerar alertas.

automação de resposta de segurança

Uma ação predefinida e programada projetada para responder ou remediar automaticamente um evento de segurança. Essas automações servem como controles de segurança responsivos ou detectivos que ajudam você a implementar as melhores práticas AWS de segurança. Exemplos de ações de resposta automatizada incluem a modificação de um grupo de segurança da VPC, a correção de uma instância EC2 da Amazon ou a rotação de credenciais.

Criptografia do lado do servidor

Criptografia dos dados em seu destino, por AWS service (Serviço da AWS) quem os recebe.

política de controle de serviços (SCP)

Uma política que fornece controle centralizado sobre as permissões de todas as contas em uma organização em AWS Organizations. SCPs defina barreiras ou estabeleça limites nas ações que um administrador pode delegar a usuários ou funções. Você pode usar SCPs como listas de permissão ou listas de negação para especificar quais serviços ou ações são permitidos ou proibidos. Para obter mais informações, consulte Políticas de controle de serviço na AWS Organizations documentação.

service endpoint (endpoint de serviço)

O URL do ponto de entrada para um AWS service (Serviço da AWS). Você pode usar o endpoint para se conectar programaticamente ao serviço de destino. Para obter mais informações, consulte Endpoints do AWS service (Serviço da AWS) na Referência geral da AWS.

acordo de serviço (SLA)

Um acordo que esclarece o que uma equipe de TI promete fornecer aos clientes, como tempo de atividade e performance do serviço.

indicador de nível de serviço (SLI)

Uma medida de um aspecto de desempenho de um serviço, como taxa de erro, disponibilidade ou taxa de transferência.

objetivo de nível de serviço (SLO)

Uma métrica alvo que representa a integridade de um serviço, conforme medida por um indicador de nível de serviço.

modelo de responsabilidade compartilhada

Um modelo que descreve a responsabilidade com a qual você compartilha AWS pela segurança e conformidade na nuvem. AWS é responsável pela segurança da nuvem, enquanto você é responsável pela segurança na nuvem. Para obter mais informações, consulte o Modelo de responsabilidade compartilhada.

SIEM

Veja informações de segurança e sistema de gerenciamento de eventos.

ponto único de falha (SPOF)

Uma falha em um único componente crítico de um aplicativo que pode interromper o sistema.

SLA

Veja o contrato de nível de serviço.

ESGUIO

Veja o indicador de nível de serviço.

SLO

Veja o objetivo do nível de serviço.

split-and-seed modelo

Um padrão para escalar e acelerar projetos de modernização. À medida que novos recursos e lançamentos de produtos são definidos, a equipe principal se divide para criar novas equipes de produtos. Isso ajuda a escalar os recursos e os serviços da sua organização, melhora a produtividade do desenvolvedor e possibilita inovações rápidas. Para obter mais informações, consulte Abordagem em fases para modernizar aplicativos no. Nuvem AWS

CUSPE

Veja um único ponto de falha.

esquema de estrelas

Uma estrutura organizacional de banco de dados que usa uma grande tabela de fatos para armazenar dados transacionais ou medidos e usa uma ou mais tabelas dimensionais menores para armazenar atributos de dados. Essa estrutura foi projetada para uso em um data warehouse ou para fins de inteligência comercial.

padrão strangler fig

Uma abordagem à modernização de sistemas monolíticos que consiste em reescrever e substituir incrementalmente a funcionalidade do sistema até que o sistema herdado possa ser desativado. Esse padrão usa a analogia de uma videira que cresce e se torna uma árvore estabelecida e, eventualmente, supera e substitui sua hospedeira. O padrão foi apresentado por Martin Fowler como forma de gerenciar riscos ao reescrever sistemas monolíticos. Para ver um exemplo de como aplicar esse padrão, consulte Modernizar incrementalmente os serviços Web herdados do Microsoft ASP.NET (ASMX) usando contêineres e o Amazon API Gateway.

sub-rede

Um intervalo de endereços IP na VPC. Cada sub-rede fica alocada em uma única zona de disponibilidade.

controle de supervisão e aquisição de dados (SCADA)

Na manufatura, um sistema que usa hardware e software para monitorar ativos físicos e operações de produção.

symmetric encryption (criptografia simétrica)

Um algoritmo de criptografia que usa a mesma chave para criptografar e descriptografar dados.

testes sintéticos

Testar um sistema de forma que simule as interações do usuário para detectar possíveis problemas ou monitorar o desempenho. Você pode usar o Amazon CloudWatch Synthetics para criar esses testes.

prompt do sistema

Uma técnica para fornecer contexto, instruções ou diretrizes a um LLM para direcionar seu comportamento. Os prompts do sistema ajudam a definir o contexto e estabelecer regras para interações com os usuários.

T

tags

Pares de valores-chave que atuam como metadados para organizar seus recursos. AWS As tags podem ajudar você a gerenciar, identificar, organizar, pesquisar e filtrar recursos. Para obter mais informações, consulte Marcar seus recursos do AWS.

variável-alvo

O valor que você está tentando prever no ML supervisionado. Ela também é conhecida como variável de resultado. Por exemplo, em uma configuração de fabricação, a variável-alvo pode ser um defeito do produto.

lista de tarefas

Uma ferramenta usada para monitorar o progresso por meio de um runbook. Uma lista de tarefas contém uma visão geral do runbook e uma lista de tarefas gerais a serem concluídas. Para cada tarefa geral, ela inclui o tempo estimado necessário, o proprietário e o progresso.

ambiente de teste

Veja o ambiente.

treinamento

O processo de fornecer dados para que seu modelo de ML aprenda. Os dados de treinamento devem conter a resposta correta. O algoritmo de aprendizado descobre padrões nos dados de treinamento que mapeiam os atributos dos dados de entrada no destino (a resposta que você deseja prever). Ele gera um modelo de ML que captura esses padrões. Você pode usar o modelo de ML para obter previsões de novos dados cujo destino você não conhece.

gateway de trânsito

Um hub de trânsito de rede que você pode usar para interconectar sua rede com VPCs a rede local. Para obter mais informações, consulte O que é um gateway de trânsito na AWS Transit Gateway documentação.

fluxo de trabalho baseado em troncos

Uma abordagem na qual os desenvolvedores criam e testam recursos localmente em uma ramificação de recursos e, em seguida, mesclam essas alterações na ramificação principal. A ramificação principal é então criada para os ambientes de desenvolvimento, pré-produção e produção, sequencialmente.

Acesso confiável

Conceder permissões a um serviço que você especifica para realizar tarefas em sua organização AWS Organizations e em suas contas em seu nome. O serviço confiável cria um perfil vinculado ao serviço em cada conta, quando esse perfil é necessário, para realizar tarefas de gerenciamento para você. Para obter mais informações, consulte Usando AWS Organizations com outros AWS serviços na AWS Organizations documentação.

tuning (ajustar)

Alterar aspectos do processo de treinamento para melhorar a precisão do modelo de ML. Por exemplo, você pode treinar o modelo de ML gerando um conjunto de rótulos, adicionando rótulos e repetindo essas etapas várias vezes em configurações diferentes para otimizar o modelo.

equipe de duas pizzas

Uma pequena DevOps equipe que você pode alimentar com duas pizzas. Uma equipe de duas pizzas garante a melhor oportunidade possível de colaboração no desenvolvimento de software.

U

incerteza

Um conceito que se refere a informações imprecisas, incompletas ou desconhecidas que podem minar a confiabilidade dos modelos preditivos de ML. Há dois tipos de incertezas: a incerteza epistêmica é causada por dados limitados e incompletos, enquanto a incerteza aleatória é causada pelo ruído e pela aleatoriedade inerentes aos dados. Para obter mais informações, consulte o guia Como quantificar a incerteza em sistemas de aprendizado profundo.

tarefas indiferenciadas

Também conhecido como trabalho pesado, trabalho necessário para criar e operar um aplicativo, mas que não fornece valor direto ao usuário final nem oferece vantagem competitiva. Exemplos de tarefas indiferenciadas incluem aquisição, manutenção e planejamento de capacidade.

ambientes superiores

Veja o ambiente.

V

aspiração

Uma operação de manutenção de banco de dados que envolve limpeza após atualizações incrementais para recuperar armazenamento e melhorar a performance.

controle de versões

Processos e ferramentas que rastreiam mudanças, como alterações no código-fonte em um repositório.

emparelhamento da VPC

Uma conexão entre duas VPCs que permite rotear o tráfego usando endereços IP privados. Para ter mais informações, consulte O que é emparelhamento de VPC? na documentação da Amazon VPC.

Vulnerabilidade

Uma falha de software ou hardware que compromete a segurança do sistema.

W

cache quente

Um cache de buffer que contém dados atuais e relevantes que são acessados com frequência. A instância do banco de dados pode ler do cache do buffer, o que é mais rápido do que ler da memória principal ou do disco.

dados mornos

Dados acessados raramente. Ao consultar esse tipo de dados, consultas moderadamente lentas geralmente são aceitáveis.

função de janela

Uma função SQL que executa um cálculo em um grupo de linhas que se relacionam de alguma forma com o registro atual. As funções de janela são úteis para processar tarefas, como calcular uma média móvel ou acessar o valor das linhas com base na posição relativa da linha atual.

workload

Uma coleção de códigos e recursos que geram valor empresarial, como uma aplicação voltada para o cliente ou um processo de back-end.

workstreams

Grupos funcionais em um projeto de migração que são responsáveis por um conjunto específico de tarefas. Cada workstream é independente, mas oferece suporte aos outros workstreams do projeto. Por exemplo, o workstream de portfólio é responsável por priorizar aplicações, planejar ondas e coletar metadados de migração. O workstream de portfólio entrega esses ativos ao workstream de migração, que então migra os servidores e as aplicações.

MINHOCA

Veja escrever uma vez, ler muitas.

WQF

Consulte Estrutura de qualificação AWS da carga de trabalho.

escreva uma vez, leia muitas (WORM)

Um modelo de armazenamento que grava dados uma única vez e evita que os dados sejam excluídos ou modificados. Os usuários autorizados podem ler os dados quantas vezes forem necessárias, mas não podem alterá-los. Essa infraestrutura de armazenamento de dados é considerada imutável.

Z

exploração de dia zero

Um ataque, geralmente malware, que tira proveito de uma vulnerabilidade de dia zero.

vulnerabilidade de dia zero

Uma falha ou vulnerabilidade não mitigada em um sistema de produção. Os agentes de ameaças podem usar esse tipo de vulnerabilidade para atacar o sistema. Os desenvolvedores frequentemente ficam cientes da vulnerabilidade como resultado do ataque.

aviso de disparo zero

Fornecer a um LLM instruções para realizar uma tarefa, mas sem exemplos (fotos) que possam ajudar a orientá-la. O LLM deve usar seu conhecimento pré-treinado para lidar com a tarefa. A

eficácia da solicitação zero depende da complexidade da tarefa e da qualidade da solicitação. Veja também a solicitação de algumas fotos.

aplicação zumbi

Uma aplicação que tem um uso médio de CPU e memória inferior a 5%. Em um projeto de migração, é comum retirar essas aplicações.

As traduções são geradas por tradução automática. Em caso de conflito entre o conteúdo da tradução e da versão original em inglês, a versão em inglês prevalecerá.