Décomposition de la base de données sur AWS

AWS Conseils prescriptifs

AWS Conseils prescriptifs: Décomposition de la base de données sur AWS

Table of Contents

Décomposition de la base de données sur AWS

Philippe Wanner et Saurabh Sharma, Amazon Web Services

octobre 2025 (historique du document)

La modernisation des bases de données, en particulier la décomposition des bases de données monolithiques, est un axe de travail essentiel pour les entreprises qui souhaitent améliorer l'agilité, l'évolutivité et les performances de leurs systèmes de gestion des données. À mesure que les entreprises se développent et que leurs besoins en données se complexifient, les bases de données monolithiques traditionnelles ont souvent du mal à suivre le rythme. Cela entraîne des problèmes de performance, des problèmes de maintenance et des difficultés d'adaptation aux exigences changeantes de l'entreprise.

Les problèmes courants liés aux bases de données monolithiques sont les suivants :

- Désalignement des domaines commerciaux — Les bases de données monolithiques ne parviennent souvent pas à aligner la technologie sur des domaines commerciaux distincts, ce qui peut limiter la croissance organisationnelle.
- Contraintes d'évolutivité — Les systèmes atteignent souvent des limites d'évolutivité, ce qui crée des obstacles à l'expansion des activités.
- Rigidité architecturale — Les structures étroitement couplées compliquent la mise à jour de composants spécifiques sans affecter l'ensemble du système.
- Dégradation des performances — L'augmentation des charges de données et l'augmentation de la simultanéité des utilisateurs entraînent souvent une détérioration des performances du système.

Les avantages de la décomposition de base de données sont les suivants :

- Agilité commerciale améliorée — La décomposition permet une adaptation rapide à l'évolution des besoins de l'entreprise et permet une mise à l'échelle indépendante.
- Performances optimisées : Decomposition vous aide à créer des solutions de base de données spécialisées adaptées à des cas d'utilisation spécifiques et à dimensionner chaque base de données de manière indépendante.
- Gestion des coûts améliorée — La décomposition permet une utilisation plus efficace des ressources et réduit les coûts opérationnels.

- Options de licence flexibles — La décomposition crée des opportunités de transition entre des licences propriétaires coûteuses et des alternatives open source.
- Facilitation de l'innovation — La décomposition facilite l'adoption de bases de données spécialement conçues pour des charges de travail spécifiques.

Public visé

Ce guide aide les architectes de bases de données, les architectes de solutions cloud, les équipes de développement d'applications et les architectes d'entreprise. Il est conçu pour vous aider à décomposer les bases de données monolithiques en magasins de données alignés sur les microservices, à mettre en œuvre des architectures de base de données axées sur le domaine, à planifier des stratégies de migration de bases de données et à adapter les opérations de base de données pour répondre aux demandes commerciales croissantes. Pour comprendre les concepts et les recommandations de ce guide, vous devez connaître les principes des bases de données relationnelles et NoSQL AWS , les services de base de données gérés et les modèles d'architecture des microservices. Ce guide est destiné à aider les organisations qui en sont aux premières étapes d'un projet de décomposition de base de données.

Objectifs

Ce guide peut aider votre organisation à atteindre les objectifs suivants :

- Collectez les exigences relatives à la décomposition de votre architecture cible.
- Développer une méthodologie systématique pour évaluer les risques et communiquer.
- Créez un plan de décomposition.
- Définissez des indicateurs de réussite, des indicateurs de performance clés (KPIs), une stratégie d'atténuation et un plan de continuité des activités.
- Améliorez l'élasticité de la charge de travail afin de répondre à la demande de l'entreprise.
- Découvrez comment adopter des bases de données spécialisées pour des cas d'utilisation spécifiques, afin de favoriser l'innovation.
- Renforcez la sécurité et la gouvernance des données de votre entreprise.
- Réduisez les coûts grâce aux mesures suivantes :
  - Frais de licence réduits
  - Réduction de la dépendance vis-à-vis des fournisseurs

- Accès amélioré à un soutien communautaire plus large et à des innovations
- Possibilité de choisir différentes technologies de base de données pour différents composants
- Migration progressive, qui réduit les risques et répartit les coûts dans le temps
- Meilleure utilisation des ressources

Défis courants et gestion des responsabilités liés à la décomposition des bases de données

La décomposition des bases de données est un processus complexe qui nécessite une planification, une exécution et une gestion minutieuses. Lorsque les entreprises cherchent à moderniser leur infrastructure de données, elles sont souvent confrontées à une multitude de défis qui peuvent avoir une incidence sur le succès de leurs projets. Cette section décrit les obstacles courants et présente une approche structurée pour surmonter ces obstacles.

Défis courants

Un projet de décomposition de base de données est confronté à plusieurs défis liés aux aspects techniques, humains et commerciaux. Sur le plan technique, garantir la cohérence des données entre les systèmes distribués constitue un obstacle majeur. Cela peut également avoir un impact potentiel sur les performances et la stabilité pendant la période de transition, et vous devez vous intégrer parfaitement aux systèmes existants. Les défis liés aux personnes incluent la courbe d'apprentissage associée au nouveau système, la résistance potentielle au changement de la part des employés et la disponibilité des ressources nécessaires. D'un point de vue commercial, le projet doit faire face aux risques de dépassement des délais, aux contraintes budgétaires et au risque d'interruption des activités pendant le processus de migration.

Définition des rôles et des responsabilités

Compte tenu de ces défis complexes qui englobent les dimensions techniques, humaines et commerciales, l'établissement de rôles et de responsabilités clairs devient essentiel à la réussite du projet. Une matrice RACI (Responsible, Accountable, Consulted and Informed) fournit la structure nécessaire pour relever ces défis. Il définit explicitement qui prend les décisions, qui exécute le travail, qui fournit des informations et qui doit rester informé à chaque étape de la décomposition. Cette clarté permet d'éviter les retards causés par une prise de décision ambiguë, d'encourager l'engagement approprié des parties prenantes et de responsabiliser les principaux résultats attendus. Sans un tel cadre, les équipes peuvent être confrontées à des responsabilités qui se chevauchent, à des communications manquées et à des trajectoires d'escalade floues, des problèmes susceptibles d'exacerber les complexités techniques existantes et les défis liés à la gestion du changement tout en augmentant le risque de dépassement des délais et des budgets.

L'exemple de matrice RACI suivant est un point de départ qui peut vous aider à clarifier les rôles et responsabilités potentiels au sein de votre organisation.

Tâche ou activité	Chef de projet	Architecte	Développeur	Partie prenante
Identifier les résultats et les défis commerciaux	A/R	R	C	–
Définir le champ d'application et identifier les exigences	A	R	C	C/I
Identifier les indicateurs de réussite du projet	A	R	C	I
Création et exécution du plan de communication	A/R	C	C	I
Définition de l'architecture cible	I	A/R	C	–
Contrôle de l'accès à la base	I	A/R	R	–
Création et exécution du plan de continuité des activités	A/R	C	I	–
Analyser la cohésion et le couplage	I	A/R	R	I

Déplacer la logique métier (telle que les procédures stockées) de la base de données vers la couche application	I	A	R	–
Découplez les relations entre les tables, appelées jointures	I	A	R	–

Définition de la portée et des exigences relatives à la décomposition des bases de données

Lorsque vous définissez la portée et identifiez les exigences de votre projet de décomposition de base de données, vous devez revenir en arrière par rapport aux besoins de votre organisation. Cela nécessite une approche systématique qui équilibre la faisabilité technique avec la valeur commerciale. Cette première étape jette les bases de l'ensemble du processus et vous aide à vous assurer que les objectifs du projet correspondent aux objectifs et aux capacités de l'organisation.

Cette section contient les rubriques suivantes :

Mise en place d'un cadre d'analyse de base

La définition du périmètre commence par un flux de travail systématique qui guide l'analyse à travers quatre phases interconnectées. Cette approche globale garantit que les efforts de décomposition des bases de données sont fondés sur une compréhension approfondie des systèmes existants et des exigences opérationnelles. Les phases du cadre d'analyse de base sont les suivantes :

1. Analyse des acteurs : identifiez minutieusement tous les systèmes et applications qui interagissent avec la base de données. Cela implique de cartographier à la fois les producteurs qui effectuent des opérations d'écriture et les consommateurs qui gèrent les opérations de lecture, tout en documentant leurs modèles d'accès, leurs fréquences et leurs heures de pointe d'utilisation. Cette vue centrée sur le client vous aide à comprendre l'impact de tout changement et à identifier les chemins critiques qui nécessitent une attention particulière lors de la décomposition.
2. Analyse des activités : analysez en profondeur les opérations spécifiques effectuées par chaque acteur. Vous créez des matrices détaillées de création, de lecture, de mise à jour et de suppression (CRUD) pour chaque système et vous identifiez les tables auxquelles ils accèdent et

comment. Cette analyse vous aide à découvrir les limites naturelles de la décomposition et met en évidence les domaines dans lesquels vous pouvez simplifier l'architecture actuelle.

3. Cartographie des dépendances — Documentez les dépendances directes et indirectes entre les systèmes, en créant des visualisations claires des flux de données et des relations. Cela permet d'identifier les points de rupture potentiels et les domaines dans lesquels une planification minutieuse est nécessaire pour gagner la confiance. L'analyse prend en compte à la fois les dépendances techniques, telles que les tables partagées et les clés étrangères, et les dépendances des processus métier, telles que les séquences de flux de travail et les exigences en matière de reporting.
4. Exigences de cohérence — Examinez les besoins de cohérence de chaque opération selon des normes élevées. Déterminez quelles opérations nécessitent une cohérence immédiate, telles que les transactions financières. D'autres opérations peuvent fonctionner avec une certaine cohérence, telles que les mises à jour analytiques. Cette analyse influence directement le choix des modèles de décomposition et les décisions architecturales tout au long du projet.

Définition des limites du système pour la décomposition des bases de données

Les limites du système sont des périmètres logiques qui définissent le point de départ d'un système et la fin d'un autre, notamment la propriété des données, les modèles d'accès et les points d'intégration. Lorsque vous définissez les limites du système, faites des choix réfléchis mais décisifs qui concilient une planification complète avec les besoins pratiques de mise en œuvre. Considérez la base de données comme une unité logique susceptible de couvrir plusieurs bases de données ou schémas physiques. Cette définition des limites permet d'atteindre les objectifs essentiels suivants :

- Identifie tous les acteurs externes et leurs modèles d'interaction
- Cartographie complète les dépendances entrantes et sortantes
- Documente les contraintes techniques et opérationnelles
- Délimite clairement l'étendue de l'effort de décomposition

Prise en compte des cycles de publication

Il est essentiel de comprendre les cycles de publication pour planifier la décomposition des bases de données. Vérifiez les délais de renouvellement du système cible et de tout système dépendant. Identifiez les opportunités de changements coordonnés. Envisagez toute mise hors service planifiée

de systèmes connectés, car cela pourrait influencer votre stratégie de décomposition. Tenez compte des fenêtres de changement existantes et des contraintes de déploiement afin de minimiser les interruptions d'activité. Assurez-vous que votre plan de mise en œuvre est conforme aux calendriers de publication de tous les systèmes connectés.

Évaluation des contraintes techniques liées à la décomposition des bases de données

Avant de procéder à la décomposition de la base de données, évaluez les principales limites techniques qui façonneront votre approche de modernisation. Examinez les capacités de votre infrastructure technologique actuelle, notamment les versions de base de données, les frameworks, les exigences de performance et les accords de niveau de service. Tenez compte des mandats de sécurité et de conformité, en particulier pour les secteurs réglementés. Passez en revue les volumes de données actuels, les prévisions de croissance et les outils de migration disponibles pour prendre des décisions éclairées en matière de mise à l'échelle. Enfin, confirmez vos droits d'accès au code source et aux modifications du système, car ceux-ci détermineront les stratégies de décomposition viables.

Comprendre le contexte organisationnel

Une décomposition de base de données réussie nécessite que vous compreniez le paysage organisationnel plus large dans lequel le système fonctionne. Cartographiez les dépendances entre les services et établissez des canaux de communication clairs entre les équipes. Évaluez les capacités techniques de votre équipe et identifiez les besoins de formation ou les lacunes en matière de compétences auxquels vous devez remédier. Tenez compte des implications de la gestion du changement, notamment de la manière de gérer les transitions et de maintenir la continuité des activités. Évaluez les ressources disponibles et toutes les contraintes, telles que les limites de budget ou de personnel. Enfin, adaptez votre stratégie de décomposition aux attentes et aux priorités des parties prenantes afin de promouvoir un soutien continu tout au long du projet.

Évaluation des risques liés à la décomposition des bases de données

Une évaluation complète des risques est essentielle au succès de la décomposition de la base de données. Évaluez soigneusement les risques, tels que l'intégrité des données pendant la migration, la dégradation potentielle des performances du système, les échecs d'intégration possibles et les

failles de sécurité. Ces défis techniques doivent être mis en balance avec les risques commerciaux, notamment les perturbations opérationnelles potentielles, les limites de ressources, les retards dans les délais et les contraintes budgétaires. Pour chaque risque identifié, élaborez des stratégies d'atténuation et des plans d'urgence spécifiques afin de maintenir la dynamique du projet tout en protégeant les opérations commerciales.

Créez une matrice des risques qui évalue à la fois l'impact et la probabilité des problèmes potentiels. Travaillez avec les équipes techniques et les parties prenantes commerciales pour identifier les risques, définir des seuils d'intervention clairs et développer des stratégies d'atténuation spécifiques. Par exemple, évaluez le risque de perte de données comme un impact élevé et une faible probabilité, et cela nécessite des stratégies de sauvegarde robustes. Une dégradation mineure des performances peut avoir un impact moyen et une probabilité élevée, et elle nécessite une surveillance proactive.

Établissez des cycles réguliers d'examen des risques pour réévaluer les priorités et ajuster les plans d'atténuation au fur et à mesure de l'évolution du projet. Cette approche systématique garantit que les ressources sont concentrées sur les risques les plus critiques tout en maintenant des trajectoires d'escalade claires pour les problèmes émergents.

Définition des critères de réussite pour la décomposition des bases de données

Les critères de réussite de la décomposition des bases de données doivent être clairement définis et mesurables à travers de multiples dimensions. D'un point de vue commercial, définissez des objectifs spécifiques en matière de réduction des coûts, d'amélioration time-to-market, de disponibilité du système et de satisfaction client. Le succès technique doit être mesuré par des améliorations quantifiables des performances du système, de l'efficacité du déploiement, de la cohérence des données et de la fiabilité globale. Pour le processus de migration, définissez des exigences strictes en matière d'absence de perte de données, de limites acceptables d'interruption des activités, de conformité budgétaire et de respect des délais.

Documentez soigneusement ces critères en maintenant des indicateurs de référence et cibles, des méthodologies de mesure claires et des calendriers de révision réguliers. Attribuez des propriétaires clairs à chaque indicateur de succès et cartographiez les dépendances entre les différents indicateurs. Cette approche globale de mesure du succès aligne les réalisations techniques sur les résultats commerciaux, tout en maintenant la responsabilité tout au long du processus de décomposition.

Contrôle de l'accès aux bases de données pendant la décomposition

De nombreuses organisations sont confrontées à un scénario commun : une base de données centrale qui s'est développée de manière organique au fil des années et à laquelle plusieurs services et équipes ont directement accès. Cela crée plusieurs problèmes critiques :

- Croissance incontrôlée — Alors que les équipes ajoutent continuellement de nouvelles fonctionnalités et modifient les schémas, la base de données devient de plus en plus complexe et difficile à gérer.
- Problèmes de performances — Même avec des améliorations matérielles, la charge croissante risque de dépasser les capacités de la base de données. Impossibilité de régler les requêtes en raison de la complexité du schéma ou du manque de compétences. Impossible de prévoir ou d'expliquer les performances du système.
- Paralysie de décomposition — Il devient presque impossible de diviser ou de refactoriser la base de données alors qu'elle est activement modifiée par plusieurs équipes.

> **Note**
>
> Les systèmes de base de données monolithiques réutilisent souvent les mêmes informations d'identification pour les applications, les services ou pour l'administration. Cela entraîne une mauvaise traçabilité des bases de données. La définition de rôles dédiés et l'adoption du principe du moindre privilège peuvent vous aider à améliorer la sécurité et la disponibilité.

Lorsqu'il s'agit d'une base de données monolithique devenue encombrante, l'un des modèles les plus efficaces pour contrôler l'accès est appelé service d'encapsulation de base de données. Il constitue une première étape stratégique dans la gestion de systèmes de bases de données complexes. Il établit un accès contrôlé aux bases de données et permet une modernisation progressive, tout en réduisant les risques. Cette approche jette les bases d'améliorations progressives en fournissant une visibilité claire sur les modèles d'utilisation des données et les dépendances. Il s'agit d'une architecture de transition qui constitue une étape vers la décomposition complète de la base de données. Le service d'emballage fournit la stabilité et le contrôle nécessaires pour réussir ce voyage.

Cette section contient les rubriques suivantes :

Contrôle de l'accès à l'aide du modèle de service d'encapsulation de base de données

Un service wrapper est une couche de service qui sert de façade à la base de données. Cette approche est particulièrement utile lorsque vous devez conserver les fonctionnalités existantes tout en préparant une future décomposition. Ce schéma suit un principe simple : lorsque quelque chose est trop salissant, commencez par le contenir. Le service wrapper devient le seul moyen autorisé d'accéder à la base de données, fournissant une interface contrôlée tout en masquant la complexité sous-jacente.

Utilisez ce modèle lorsque la décomposition immédiate de la base de données n'est pas possible en raison de schémas complexes ou lorsque plusieurs services nécessitent un accès continu aux données. Il est particulièrement utile pendant les périodes de transition, car il permet de procéder à une refactorisation minutieuse tout en préservant la stabilité du système. Ce modèle fonctionne bien lors de la consolidation de la propriété des données entre les mains d'équipes spécifiques ou lorsque les nouvelles applications nécessitent des vues agrégées sur plusieurs tables.

Par exemple, appliquez ce modèle lorsque :

- La complexité du schéma empêche une séparation immédiate
- Plusieurs équipes ont besoin d'un accès permanent aux données
- Une modernisation progressive est préférable
- La restructuration de l'équipe nécessite une propriété claire des données
- Les nouvelles applications ont besoin de vues de données consolidées

Avantages et limites du modèle de service d'encapsulation de base de données

Les avantages du modèle d'enveloppe de base de données sont les suivants :

- Croissance contrôlée — Le service wrapper empêche d'autres ajouts incontrôlés au schéma de base de données.

- Limites claires — Le processus de mise en œuvre vous aide à établir des limites claires en matière de propriété et de responsabilité.
- Liberté de refactorisation — Un service d'emballage vous permet d'apporter des modifications internes sans affecter les consommateurs.
- Observabilité améliorée — Un service d'encapsulation est un point unique de surveillance et de journalisation.
- Tests simplifiés — Un service d'encapsulation permet aux utilisateurs de services de créer plus facilement des versions fictives simplifiées à des fins de test.

Les limites du modèle d'enveloppe de base de données sont les suivantes.

- Couplage technologique — Un service d'encapsulation fonctionne mieux lorsqu'il utilise la même pile technologique que les services consommateurs.
- Frais généraux initiaux — Le service d'encapsulation nécessite une infrastructure supplémentaire susceptible d'affecter les performances.
- Effort de migration — Pour mettre en œuvre le service d'encapsulation, vous devez coordonner l'ensemble des équipes afin de passer à l'abandon de l'accès direct.
- Performances — Si le service d'encapsulage est soumis à un trafic élevé, à une utilisation intensive ou à un accès fréquent, les services consommateurs risquent de connaître des performances médiocres. En plus de la base de données, le service wrapper doit gérer la pagination, les curseurs et les connexions à la base de données. Selon votre cas d'utilisation, il se peut qu'il ne soit pas bien évolutif et qu'il ne soit pas adapté aux charges de travail d'extraction, de transformation et de chargement (ETL).

Implémentation du modèle de service d'encapsulation de base de données

La mise en œuvre du modèle de service d'encapsulation de base de données comporte deux phases. Vous devez d'abord créer le service d'encapsulation de base de données. Ensuite, vous dirigez tous les accès par son intermédiaire et vous documentez les modèles d'accès.

Phase 1 : Création du service d'encapsulation de base de données

Créez une couche de service légère qui agit comme un gardien d'accès à votre base de données. Dans un premier temps, il devrait refléter toutes les fonctionnalités existantes. Ce service wrapper devient le point d'accès obligatoire pour toutes les opérations de base de données, ce qui convertit les dépendances directes de la base de données en dépendances au niveau du service. Mettez

en œuvre une journalisation et une surveillance détaillées au niveau de cette couche pour suivre les modèles d'utilisation, les indicateurs de performance et les fréquences d'accès. Conservez vos procédures stockées existantes, mais assurez-vous qu'elles ne sont accessibles que via cette nouvelle interface de service.

Phase 2 : Mise en œuvre du contrôle d'accès

Redirigez systématiquement tous les accès à la base de données via le service wrapper, puis révoquez les autorisations directes de base de données des systèmes externes qui accèdent directement à la base de données. Documentez chaque modèle d'accès et chaque dépendance au fur et à mesure que les services sont migrés. Cet accès contrôlé permet de refactoriser en interne les composants de la base de données sans perturber les consommateurs externes. Par exemple, commencez par des opérations en lecture seule à faible risque plutôt que par des flux de travail transactionnels complexes.

Phase 3 : surveillance des performances de la base de données

Utilisez le service Wrapper comme point de surveillance centralisé pour les performances de la base de données. Suivez les indicateurs clés, notamment les temps de réponse aux requêtes, les modèles d'utilisation, les taux d'erreur et l'utilisation des ressources. Configurez des alertes pour les seuils de performance et les modèles inhabituels. Par exemple, surveillez les requêtes lentes, l'utilisation du pool de connexions et le débit des transactions pour identifier de manière proactive les problèmes potentiels.

Utilisez cette vue consolidée pour optimiser les performances de la base de données grâce au réglage des requêtes, aux ajustements de l'allocation des ressources et à l'analyse des modèles d'utilisation. La nature centralisée du service d'emballage facilite la mise en œuvre des améliorations et la validation de leur impact auprès de tous les consommateurs, tout en maintenant des normes de performance cohérentes.

Bonnes pratiques pour la mise en œuvre d'un service d'encapsulation de base de données

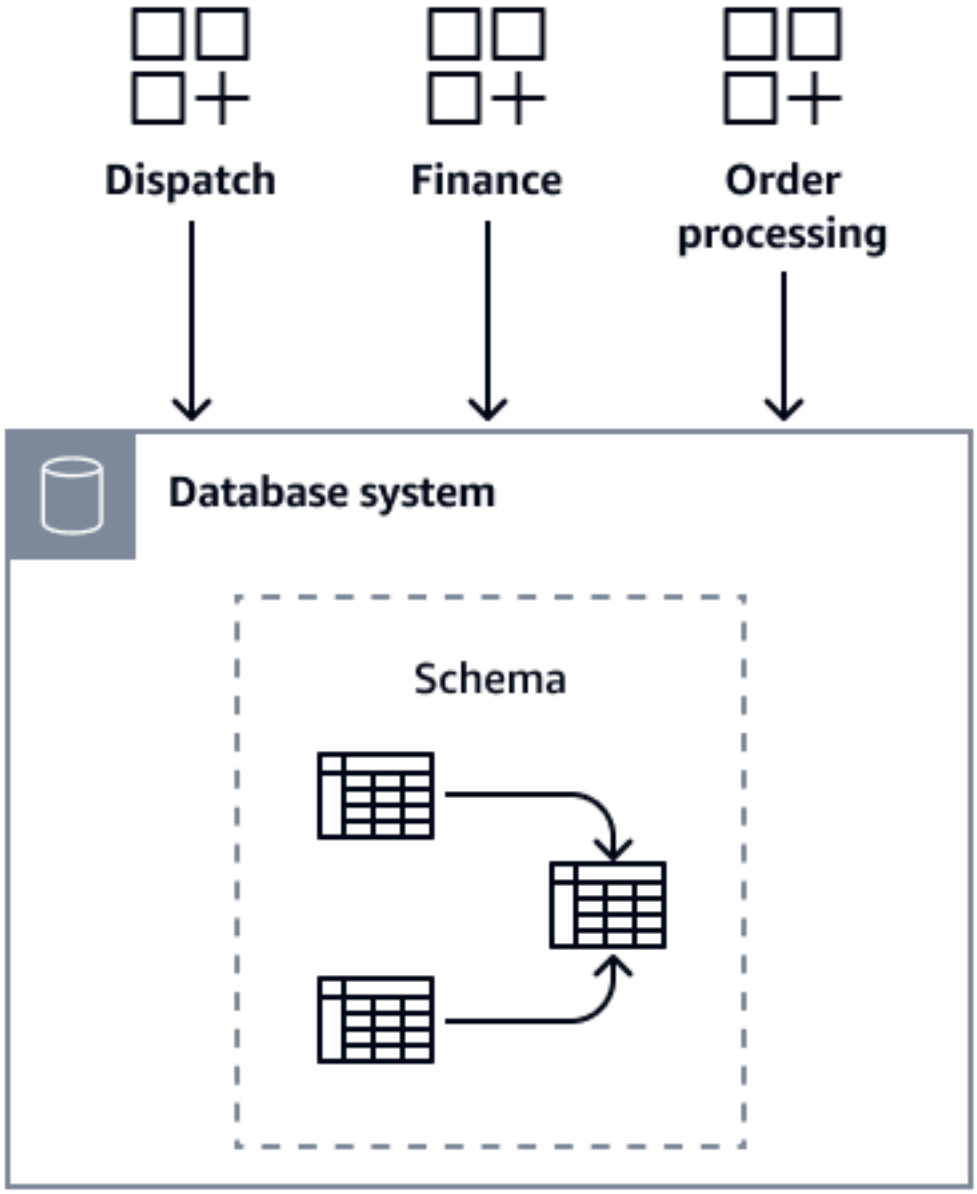
Les meilleures pratiques suivantes peuvent vous aider à implémenter un service d'encapsulation de base de données :

- Commencez modestement : commencez par un wrapper minimal qui se contente de reproduire les fonctionnalités existantes

- Maintien de la stabilité : maintien de la stabilité de l'interface de service tout en apportant des améliorations internes
- Surveiller l'utilisation : mettre en œuvre une surveillance complète pour comprendre les modèles d'accès
- Responsabilité claire — Désignez une équipe dédiée chargée de maintenir à la fois le wrapper et le schéma sous-jacent
- Encourager le stockage local — Motivez les équipes à stocker leurs données dans leurs propres bases de données

Exemple basé sur un scénario

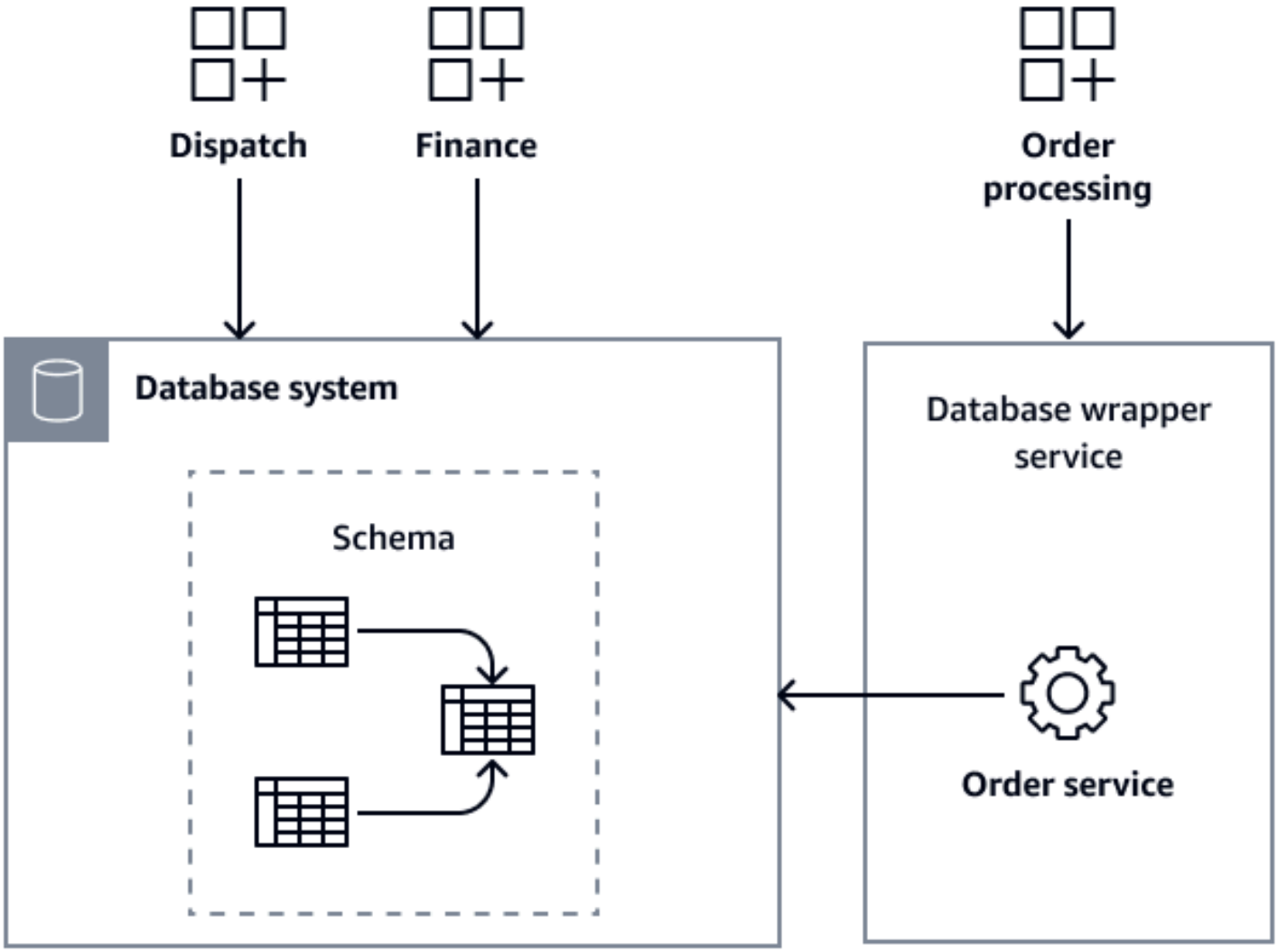
Cette section décrit un exemple de la façon dont une société fictive, nommée AnyCompany Books, pourrait utiliser le modèle d'encapsulation de base de données pour contrôler l'accès à son système de base de données monolithique. Chez AnyCompany Books, il existe trois services essentiels : l'expédition, les finances et le traitement des commandes. Ces services partagent l'accès à une base de données centrale. Chaque service est géré par une équipe différente. Au fil du temps, ils modifient indépendamment le schéma de base de données pour répondre à leurs besoins spécifiques. Cela a conduit à un enchevêtrement de dépendances et à une structure de base de données de plus en plus complexe.



L'architecte d'application ou d'entreprise de l'entreprise reconnaît la nécessité de décomposer cette base de données monolithique. Leur objectif est de doter chaque service de sa propre base de données dédiée afin d'améliorer la maintenabilité et de réduire les dépendances entre les équipes. Cependant, elles sont confrontées à un défi de taille : il est presque impossible de décomposer la base de données alors que les trois équipes continuent de la modifier activement pour leurs projets en cours. Les changements constants de schéma et le manque de coordination entre les équipes font qu'il est extrêmement risqué de tenter une restructuration significative.

L'architecte utilise le modèle de service d'encapsulation de base de données pour commencer à contrôler l'accès à la base de données monolithique. Tout d'abord, ils ont configuré le service

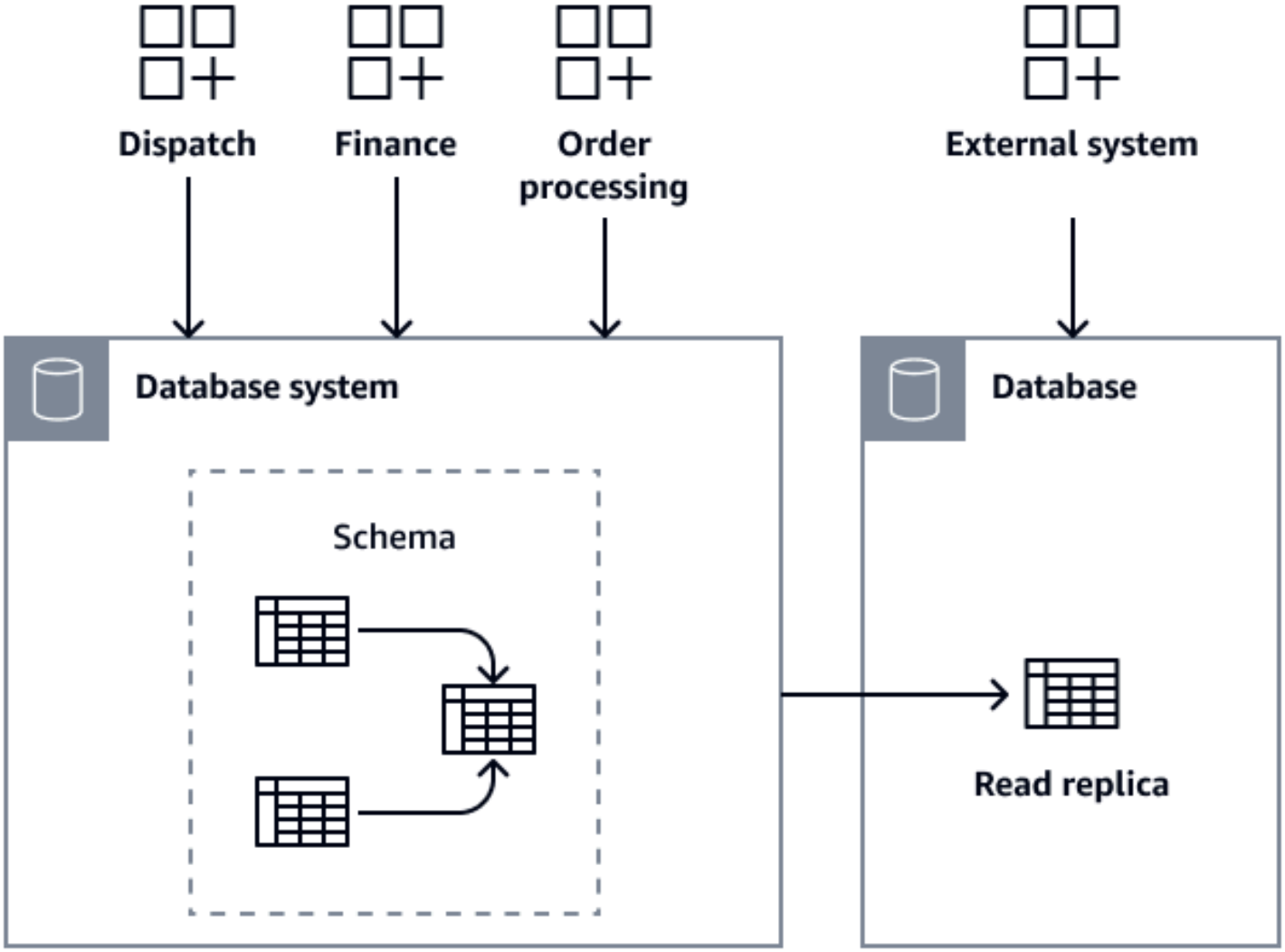
d'encapsulation de base de données pour un module particulier, appelé service Order. Ensuite, ils redirigent le service de traitement des commandes pour accéder au service wrapper au lieu d'accéder directement à la base de données. L'image suivante montre l'infrastructure modifiée.



Contrôle de l'accès avec le modèle CQRS

Un autre modèle que vous pouvez utiliser pour isoler les systèmes externes qui se connectent à cette base de données centrale est la ségrégation des responsabilités des requêtes de commande (CQRS). Si certains systèmes externes se connectent à votre base de données centrale principalement à des fins de lecture, telles que des analyses, des rapports ou d'autres opérations intensives en lecture, vous pouvez créer des magasins de données distincts optimisés pour la lecture.

Ce modèle isole efficacement ces systèmes externes des impacts de la décomposition des bases de données et des modifications de schéma. En conservant des répliques de lecture dédiées ou des magasins de données spécialement conçus pour des modèles de requêtes spécifiques, les équipes peuvent poursuivre leurs opérations sans être affectées par les modifications de la structure de base de données principale. Par exemple, pendant que vous décomposez votre base de données monolithique, les systèmes de reporting peuvent continuer à fonctionner avec leurs vues de données existantes, et les charges de travail analytiques peuvent conserver leurs modèles de requêtes actuels grâce à des magasins d'analyse dédiés. Cette approche fournit une isolation technique et favorise l'autonomie organisationnelle, car les différentes équipes peuvent faire évoluer leurs systèmes indépendamment, sans lien étroit avec le processus de transformation de la base de données principale.



Pour plus d'informations sur ce modèle et un exemple de son utilisation pour découpler les relations entre les tables, voir Motif CQRS plus loin dans ce guide.

Analyse de la cohésion et du couplage pour la décomposition des bases de données

Cette section vous aide à analyser les modèles de couplage et de cohésion dans votre base de données monolithique afin de guider sa décomposition. Il est essentiel de comprendre comment les composants de base de données interagissent et dépendent les uns des autres pour identifier les points de rupture naturels, évaluer la complexité et planifier une approche de migration progressive. Cette analyse révèle les dépendances cachées, met en évidence les domaines qui se prêtent à une séparation immédiate et vous aide à hiérarchiser les efforts de décomposition tout en minimisant les risques de transformation. En examinant à la fois le couplage et la cohésion, vous pouvez prendre des décisions éclairées concernant la séquence de séparation des composants afin de maintenir la stabilité du système tout au long du processus de transformation.

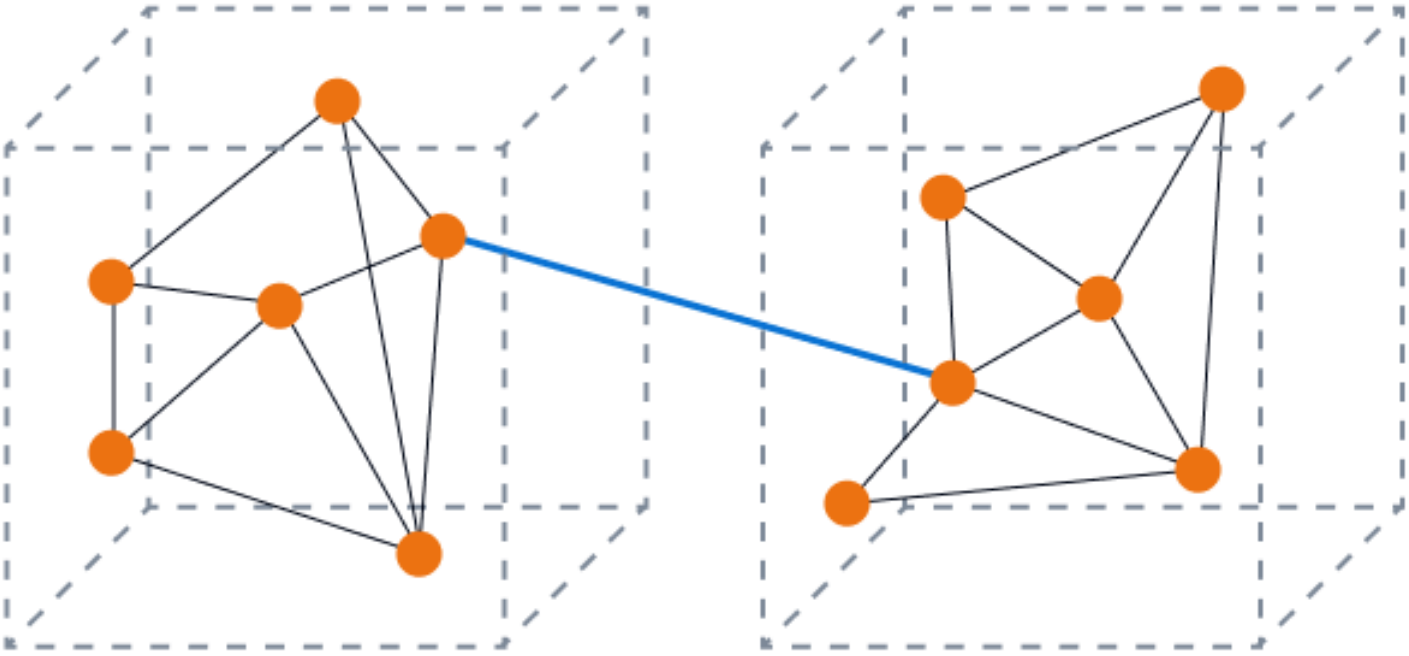
Cette section contient les rubriques suivantes :

- À propos de la cohésion et du couplage
- Modèles de couplage courants dans les bases de données monolithiques
- Modèles de cohésion courants dans les bases de données monolithiques
- Mise en œuvre d'un faible couplage et d'une haute cohésion

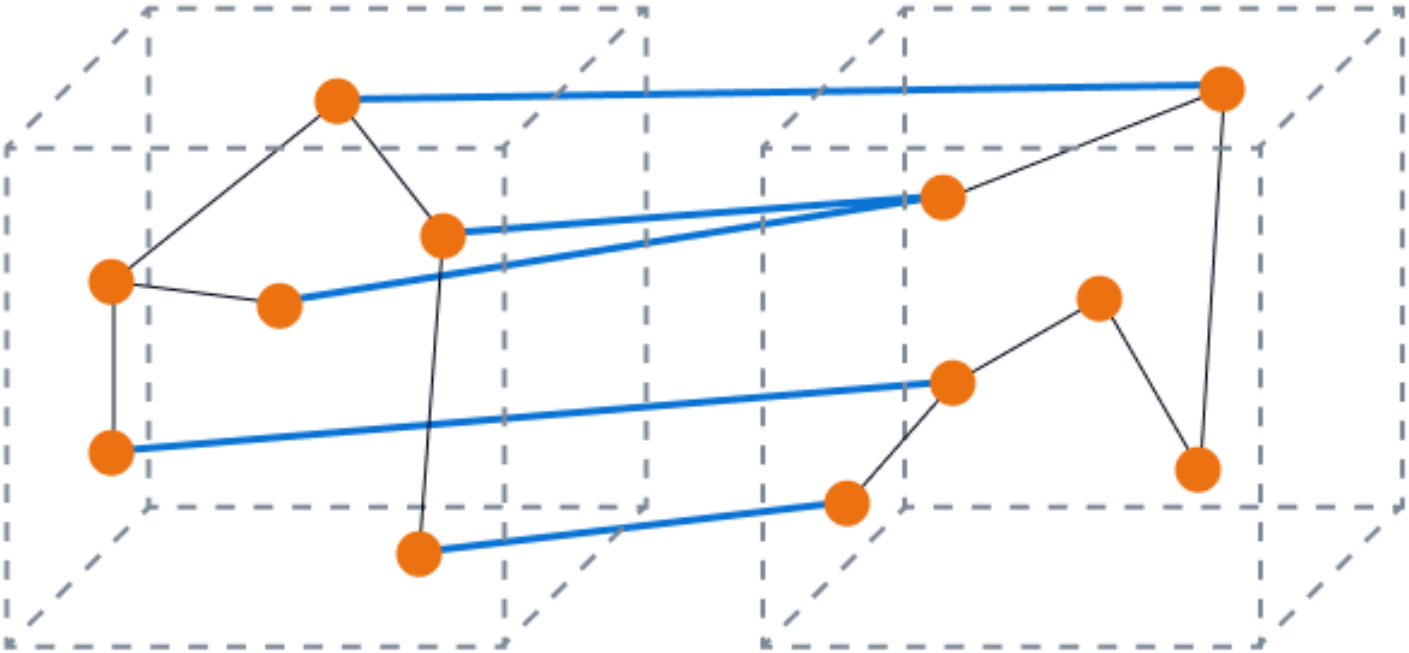
À propos de la cohésion et du couplage

Le couplage mesure le degré d'interdépendance entre les composants de la base de données. Dans un système bien conçu, vous souhaitez obtenir un couplage souple, dans lequel les modifications apportées à un composant ont un impact minimal sur les autres. La cohésion mesure dans quelle mesure les éléments d'un composant de base de données fonctionnent ensemble pour atteindre un objectif unique et bien défini. Une cohésion élevée indique que les éléments d'un composant sont étroitement liés et axés sur une fonction spécifique. Lorsque vous décomposez une base de données monolithique, vous devez analyser à la fois la cohésion au sein des composants individuels et le couplage entre eux. Cette analyse vous aide à prendre des décisions éclairées sur la manière de décomposer la base de données tout en préservant l'intégrité et les performances du système.

L'image suivante montre un couplage lâche avec une cohésion élevée. Les composants de la base de données fonctionnent ensemble pour exécuter une fonction spécifique, et vous minimisez l'impact des modifications sur un seul composant. C'est l'état idéal.



L'image suivante montre un couplage élevé avec une faible cohésion. Les composants de la base de données sont déconnectés et les modifications sont très susceptibles d'avoir un impact sur les autres composants.



Modèles de couplage courants dans les bases de données monolithiques

Il existe plusieurs modèles de couplage courants lors de la décomposition d'une base de données monolithique en bases de données spécifiques aux microservices. La compréhension de ces modèles est essentielle au succès des initiatives de modernisation des bases de données. Cette section décrit chaque modèle, ses défis et les meilleures pratiques pour réduire le couplage.

Modèle de couplage de mise en œuvre

Définition : Les composants sont étroitement interconnectés au niveau du code et du schéma. Par exemple, la modification de la structure d'une `customer` table a un impact sur `order` les `billing` services. `inventory`

Impact de la modernisation : chaque microservice nécessite son propre schéma de base de données et sa propre couche d'accès aux données.

Défis :

- Les modifications apportées aux tables partagées affectent plusieurs services
- Risque élevé d'effets secondaires imprévus
- Complexité accrue des tests
- Difficile de modifier des composants individuels

Bonnes pratiques pour réduire le couplage :

- Définissez des interfaces claires entre les composants
- Utiliser des couches d'abstraction pour masquer les détails de mise en œuvre
- Implémenter des schémas spécifiques au domaine

Schéma de couplage temporel

Définition : Les opérations doivent s'exécuter dans un ordre spécifique. Par exemple, le traitement des commandes ne peut pas se poursuivre tant que les mises à jour de l'inventaire ne sont pas terminées.

Impact de la modernisation : chaque microservice a besoin d'un contrôle autonome des données.

Défis :

- Rompre les dépendances synchrones entre les services
- Les goulets d'étranglement liés aux performances
- Difficile à optimiser
- Traitement parallèle limité

Bonnes pratiques pour réduire le couplage :

- Mettre en œuvre un traitement asynchrone dans la mesure du possible
- Utiliser des architectures pilotées par les événements
- Conception pour une cohérence éventuelle, le cas échéant

Schéma de couplage de déploiement

Définition : Les composants du système doivent être déployés en tant qu'unité unique. Par exemple, une modification mineure de la logique de traitement des paiements nécessite le redéploiement de l'ensemble de la base de données.

Impact de la modernisation : déploiements de bases de données indépendants par service

Défis :

- Déploiements à haut risque
- Fréquence de déploiement limitée
- Procédures de rétrogradation complexes

Bonnes pratiques pour réduire le couplage :

- Décomposer en composants déployables indépendamment
- Mettre en œuvre des stratégies de partage de base de données
- Utiliser des modèles de déploiement bleu-vert

Schéma de couplage de domaines

Définition : Les domaines commerciaux partagent les structures et la logique des bases de données. Par exemple, les `inventory` domaines `customerorder`, et partagent des tables et des procédures stockées.

Impact de la modernisation : isolation des données spécifiques au domaine

Défis :

- Limites de domaines complexes
- Difficile de faire évoluer des domaines individuels
- Des règles commerciales enchevêtrées

Bonnes pratiques pour réduire le couplage :

- Identifiez des limites de domaine claires
- Séparer les données par contexte de domaine
- Mettre en œuvre des services spécifiques à un domaine

Modèles de cohésion courants dans les bases de données monolithiques

Il existe plusieurs modèles de cohésion courants lors de l'évaluation des composants de base de données en vue de leur décomposition. Il est essentiel de comprendre ces modèles pour identifier les composants de base de données bien structurés. Cette section décrit chaque modèle, ses caractéristiques et les meilleures pratiques pour renforcer la cohésion.

Schéma de cohésion fonctionnel

Définition : Tous les éléments soutiennent et contribuent directement à l'exécution d'une fonction unique et bien définie. Par exemple, toutes les procédures et tables stockées dans un module de traitement des paiements ne gèrent que les opérations liées au paiement.

Impact de la modernisation : modèle idéal pour la conception de bases de données de microservices

Défis :

- Identifier des limites fonctionnelles claires
- Séparer les composants à usage mixte
- Maintien d'une responsabilité unique

Les meilleures pratiques pour renforcer la cohésion :

- Regroupez les fonctions associées
- Supprimer les fonctionnalités non liées
- Définissez des limites claires pour les composants

Schéma de cohésion séquentiel

Définition : La sortie d'un élément devient l'entrée d'un autre. Par exemple, les résultats de validation d'une commande sont intégrés au traitement des commandes.

Impact de la modernisation : nécessite une analyse minutieuse des flux de travail et une cartographie des flux de données

Défis :

- Gérer les dépendances entre les étapes
- Gestion des scénarios de défaillance
- Maintien de l'ordre des processus

Les meilleures pratiques pour renforcer la cohésion :

- Documentez des flux de données clairs
- Mettre en œuvre une gestion appropriée des erreurs
- Concevez des interfaces claires entre les étapes

Schéma de cohésion communicationnelle

Définition : Les éléments fonctionnent sur les mêmes données. Par exemple, les fonctions de gestion des profils clients fonctionnent toutes avec les données des clients.

Impact de la modernisation : aide à identifier les limites des données pour la séparation des services afin de réduire le couplage entre les modules

Défis :

- Déterminer la propriété des données
- Gestion de l'accès aux données partagées
- Maintien de la cohérence des données

Les meilleures pratiques pour renforcer la cohésion :

- Définissez clairement la propriété des données
- Mettre en œuvre des modèles d'accès aux données appropriés
- Concevez un partitionnement des données efficace

Schéma de cohésion procédurale

Définition : Les éléments sont regroupés car ils doivent être exécutés dans un ordre spécifique, mais ils peuvent ne pas être liés fonctionnellement. Par exemple, dans le cadre du traitement des commandes, une procédure stockée qui gère à la fois la validation des commandes et les notifications aux utilisateurs est regroupée simplement parce qu'elles se déroulent en séquence, même si elles répondent à des objectifs différents et peuvent être gérées par des services distincts.

Impact de la modernisation : nécessite une séparation soigneuse des procédures tout en maintenant le flux des processus

Défis :

- Maintien d'un flux de processus correct après la décomposition
- Identifier les véritables limites fonctionnelles par rapport aux dépendances procédurales

Les meilleures pratiques pour renforcer la cohésion :

- Procédures distinctes en fonction de leur objectif fonctionnel plutôt que de leur ordre d'exécution
- Utiliser des modèles d'orchestration pour gérer le flux de processus
- Mettre en œuvre des systèmes de gestion des flux de travail pour les séquences complexes

- Concevez des architectures pilotées par les événements pour gérer les étapes du processus de manière indépendante

Schéma de cohésion temporelle

Définition : Les éléments sont liés par des exigences temporelles. Par exemple, lorsqu'une commande est passée, plusieurs opérations doivent être exécutées simultanément : la vérification de l'inventaire, le traitement des paiements, la confirmation de commande et la notification d'expédition doivent tous avoir lieu dans un intervalle de temps spécifique afin de maintenir un état de commande cohérent.

Impact de la modernisation : peut nécessiter un traitement spécial dans les systèmes distribués

Défis :

- Coordination des dépendances temporelles entre les services distribués
- Gestion des transactions distribuées
- Confirmation de l'achèvement du processus sur plusieurs composants

Les meilleures pratiques pour renforcer la cohésion :

- Mettre en œuvre des mécanismes de planification et des délais d'attente appropriés
- Utilisez des architectures axées sur les événements avec une gestion claire des séquences
- Conception pour une cohérence éventuelle avec les modèles de rémunération
- Implémenter des modèles de saga pour les transactions distribuées

Schéma de cohésion logique ou fortuit

Définition : Les éléments sont classés logiquement pour faire les mêmes choses, même s'ils ont des relations faibles ou insignifiantes. Par exemple, vous pouvez stocker les données relatives aux commandes des clients, les inventaires des entrepôts et les modèles d'e-mails marketing dans le même schéma de base de données, car ils concernent tous les opérations de vente, malgré des modèles d'accès, une gestion du cycle de vie et des exigences de mise à l'échelle différents. Un autre exemple consiste à combiner le traitement des paiements des commandes et la gestion du catalogue de produits au sein d'un même composant de base de données, car ils font tous deux

partie du système de commerce électronique, même s'ils répondent à des fonctions commerciales distinctes ayant des besoins opérationnels différents.

Impact de la modernisation : devrait être refactorisé ou réorganisé

Défis :

- Identifier de meilleurs modèles d'organisation
- Briser les dépendances inutiles
- Composantes de restructuration qui ont été regroupées de manière arbitraire

Les meilleures pratiques pour renforcer la cohésion :

- Réorganisez en fonction des véritables limites fonctionnelles et des domaines d'activité
- Supprimer les groupements arbitraires basés sur des relations superficielles
- Mettre en œuvre une séparation appropriée des éléments en fonction des capacités de l'entreprise
- Aligner les composants de base de données sur leurs exigences opérationnelles spécifiques

Mise en œuvre d'un faible couplage et d'une haute cohésion

Bonnes pratiques

Les meilleures pratiques suivantes peuvent vous aider à réduire le couplage :

- Réduisez les dépendances entre les composants de base de données
- Utiliser des interfaces bien définies pour l'interaction entre les composants
- Évitez les états partagés et les structures de données globales

Les meilleures pratiques suivantes peuvent vous aider à atteindre une cohésion élevée :

- Regroupez les données et les opérations associées
- Assurez-vous que chaque composant a une responsabilité unique et claire
- Maintenir des limites claires entre les différents domaines d'activité

Phase 1 : Cartographier les dépendances des données

Cartographiez les relations entre les données et identifiez les limites naturelles. Vous pouvez utiliser des outils, tels que SchemaSpy, pour visualiser la base de données en affichant les tables dans un diagramme entité-relation (ER). Cela fournit une analyse statique de la base de données et indique certaines des limites et dépendances claires au sein de la base de données.

Vous pouvez également exporter les schémas de votre base de données dans une base de données de graphes ou dans un Jupiter bloc-notes. Vous pouvez ensuite appliquer des algorithmes de clustering ou de composants interconnectés pour identifier les limites naturelles et les dépendances. D'autres AWS Partner outils, tels que CAST Imaging, peuvent vous aider à comprendre les dépendances de votre base de données.

Phase 2 : Analyser les limites des transactions et les modèles d'accès

Analysez les modèles de transaction pour maintenir les propriétés ACID (atomicité, cohérence, isolation, durabilité) et comprendre comment les données sont consultées et modifiées. Vous pouvez utiliser des outils d'analyse de base de données et de diagnostic, tels que Oracle Automatic Workload Repository (AWR)ou PostgreSQL pg_stat_statements. Cette analyse vous aide à comprendre qui accède à la base de données et quelles sont les limites des transactions. Cela peut également vous aider à comprendre la cohésion et le couplage entre les tables lors de l'exécution. Vous pouvez également utiliser des outils de surveillance et de profilage qui peuvent lier le code et les profils d'exécution de base de données, tels que Dynatrace AppEngine.

Les outils d'IA, tels que vFunction, peuvent vous aider à identifier les limites des domaines en analysant les limites fonctionnelles et de domaine de l'application. Bien qu'il analyse vFunction principalement la couche applicative, ses informations peuvent guider la décomposition de l'application et de la base de données, favorisant ainsi l'alignement avec les domaines commerciaux.

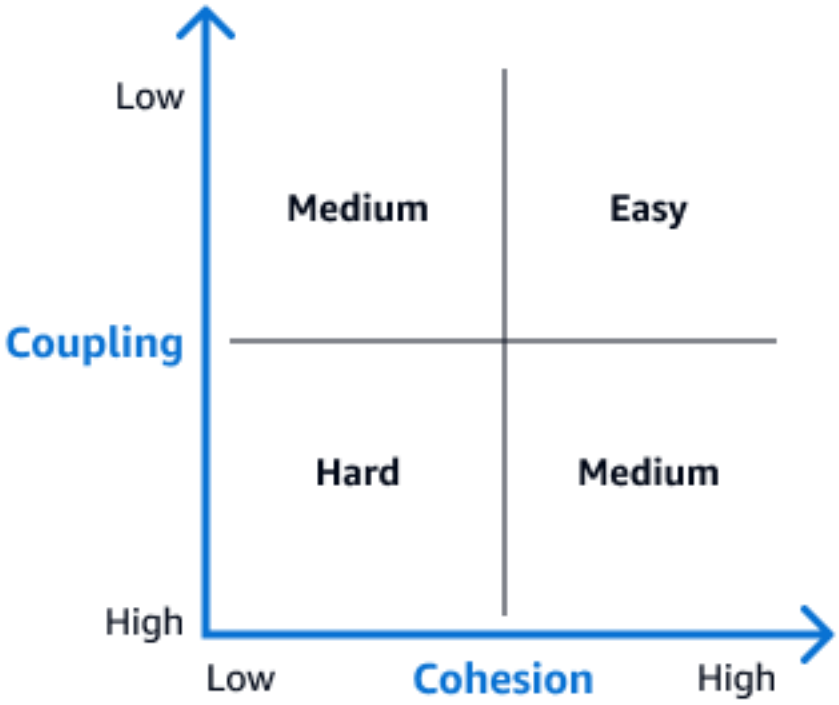
Phase 3 : Identifier les tables autonomes

Recherchez les tableaux qui présentent deux caractéristiques principales :

- Cohésion élevée — Les contenus de la table sont étroitement liés les uns aux autres
- Couplage faible : leur dépendance par rapport aux autres tables est minimale.

La matrice de couplage-cohésion suivante peut vous aider à identifier la difficulté de découpler chaque table. Les tableaux qui apparaissent dans le quadrant supérieur droit de cette matrice

sont des candidats idéaux pour les premiers efforts de découplage, car ils sont les plus faciles à séparer. Dans un diagramme ER, ces tables présentent peu de relations de clé étrangère ou d'autres dépendances. Après avoir découplé ces tables, progressez vers des tables présentant des relations plus complexes.



> **Note**
>
> La structure de la base de données reflète souvent l'architecture de l'application. Les tables plus faciles à découpler au niveau de la base de données correspondent généralement à des composants plus faciles à convertir en microservices au niveau de l'application.

Migration de la logique métier de la base de données vers la couche application

La migration de la logique métier des procédures, des déclencheurs et des fonctions stockés dans les bases de données vers les services de couche applicative est une étape essentielle de la décomposition des bases de données monolithiques. Cette transformation améliore l'autonomie des services, simplifie la maintenance et améliore l'évolutivité. Cette section fournit des conseils sur l'analyse de la logique de base de données, la planification de la stratégie de migration, puis la mise en œuvre de la transformation tout en maintenant la continuité des activités. Il traite également de l'établissement d'un plan de réduction efficace.

Cette section contient les rubriques suivantes :

- Phase 1 : Analyse de la logique métier
- Phase 2 : Classification de la logique métier
- Phase 3 : Migration de la logique métier
- Stratégie de rétrogradation pour la logique métier

Phase 1 : Analyse de la logique métier

Lorsque vous modernisez des bases de données monolithiques, vous devez d'abord effectuer une analyse complète de la logique de votre base de données existante. Cette phase se concentre sur trois catégories principales :

- Les procédures stockées contiennent souvent des opérations commerciales critiques, notamment la logique de manipulation des données, les règles métier, les contrôles de validation et les calculs. En tant que composants essentiels de la logique métier de l'application, ils nécessitent une décomposition minutieuse. Par exemple, les procédures stockées d'une organisation financière peuvent gérer le calcul des intérêts, le rapprochement des comptes et les contrôles de conformité.
- Les déclencheurs sont des composants clés de la base de données qui gèrent les pistes d'audit, la validation des données, les calculs et la cohérence entre les tables. Par exemple, une entreprise de vente au détail peut utiliser des déclencheurs pour gérer les mises à jour des stocks dans l'ensemble de son système de traitement des commandes, ce qui montre la complexité des opérations de base de données automatisées.

- Les fonctions des bases de données gèrent principalement les transformations de données, les calculs et les opérations de recherche. Ils sont souvent intégrés dans de multiples procédures et applications. Par exemple, un établissement de santé peut utiliser des fonctions pour normaliser les données des patients ou rechercher des codes médicaux.

Chaque catégorie représente différents aspects de la logique métier intégrée dans la couche de base de données. Vous devez évaluer et planifier soigneusement chacune d'elles afin de les faire migrer vers la couche application.

Au cours de cette phase d'analyse, les clients sont généralement confrontés à trois défis importants. Tout d'abord, des dépendances complexes apparaissent par le biais d'appels de procédures imbriqués, de références entre schémas et de dépendances de données implicites. Ensuite, la gestion des transactions devient essentielle, en particulier lorsqu'il s'agit de transactions en plusieurs étapes et que l'on assure la cohérence des données entre les systèmes distribués. Troisièmement, les considérations relatives aux performances doivent être soigneusement évaluées, en particulier pour les opérations de traitement par lots, les mises à jour massives des données et les calculs en temps réel qui bénéficient actuellement de la proximité des données.

Pour relever efficacement ces défis, vous pouvez utiliser AWS Schema Conversion Tool (AWS SCT) pour l'analyse initiale, puis utiliser des outils détaillés de mappage des dépendances. Cette approche vous permet de comprendre l'étendue complète de la logique de votre base de données et de créer une stratégie de migration complète qui assure la continuité des activités pendant la décomposition.

En comprenant parfaitement ces composants et ces défis, vous pouvez mieux planifier votre parcours de modernisation et prendre des décisions éclairées quant aux éléments à prioriser lors de la migration vers une architecture basée sur les microservices.

Lorsque vous analysez les composants du code de base de données, créez une documentation complète pour chaque procédure stockée, déclencheur et fonction. Commencez par décrire clairement son objectif et ses fonctionnalités de base, y compris les règles métier qu'il met en œuvre. Détaillez tous les paramètres d'entrée et de sortie, et notez leurs types de données et leurs plages valides. Cartographiez les dépendances vis-à-vis d'autres objets de base de données, de systèmes externes et de processus en aval. Définissez clairement les limites des transactions et les exigences d'isolation afin de préserver l'intégrité des données. Documentez toutes les attentes en matière de performance, y compris les exigences en matière de temps de réponse et les modèles d'utilisation des ressources. Enfin, analysez les modèles d'utilisation pour comprendre les pics de charge, la fréquence d'exécution et les périodes commerciales critiques.

Phase 2 : Classification de la logique métier

Une décomposition efficace des bases de données nécessite une catégorisation systématique de la logique de base de données selon des dimensions clés : complexité, impact commercial, dépendances, modèles d'utilisation et difficulté de migration. Cette classification vous aide à identifier les composants à haut risque, à déterminer les exigences de test et à établir les priorités de migration. Par exemple, les procédures stockées complexes ayant un impact commercial important et une utilisation fréquente nécessitent une planification minutieuse et des tests approfondis. Cependant, des fonctions simples, rarement utilisées avec des dépendances minimales peuvent convenir aux premières phases de migration.

Cette approche structurée crée une feuille de route de migration équilibrée qui minimise les interruptions d'activité tout en préservant la stabilité du système. En comprenant ces interrelations, vous pouvez améliorer la séquence de vos efforts de décomposition et allouer les ressources de manière appropriée.

Phase 3 : Migration de la logique métier

Après avoir analysé et classé votre logique métier, il est temps de la migrer. Il existe deux approches lors de la migration de la logique métier depuis une base de données monolithique : déplacer la logique de base de données vers la couche application ou déplacer la logique métier vers une autre base de données faisant partie du microservice.

Si vous migrez la logique métier vers l'application, les tables de base de données stockent uniquement les données et la base de données ne contient aucune logique métier. Il s'agit de l'approche recommandée. Vous pouvez utiliser Ispirer ou des outils d'IA générative, tels qu'Amazon Q Developer Kiro, ou pour convertir la logique métier de base de données pour la couche application, telle que la conversion en Java. Pour plus d'informations, voir Migrer la logique métier de la base de données vers l'application pour accélérer l'innovation et la flexibilité (article de AWS blog).

Si vous migrez la logique métier vers une autre base de données, vous pouvez utiliser AWS Schema Conversion Tool (AWS SCT) pour convertir les schémas de base de données et les objets de code existants vers votre base de données cible. Il prend en charge des services de AWS base de données spécialement conçus, tels qu'Amazon DynamoDB, Amazon Aurora et Amazon Redshift. En fournissant un rapport d'évaluation complet et des fonctionnalités de conversion automatisées, AWS SCT cela permet de rationaliser le processus de transition, ce qui vous permet de vous concentrer sur l'optimisation de votre nouvelle structure de base de données pour améliorer les performances et l'évolutivité. Au fur et à mesure que vous progressez dans votre projet de modernisation, vous AWS

SCT pouvez gérer des conversions incrémentielles afin de soutenir une approche progressive, vous permettant de valider et d'affiner chaque étape de la transformation de votre base de données.

Stratégie de rétrogradation pour la logique métier

Deux aspects essentiels de toute stratégie de décomposition sont le maintien de la rétrocompatibilité et la mise en œuvre de procédures de restauration complètes. Ces éléments fonctionnent ensemble pour aider à protéger les opérations pendant la période de transition. Cette section décrit comment gérer la compatibilité pendant le processus de décomposition et établir des capacités de restauration d'urgence efficaces qui protègent contre les problèmes potentiels.

Maintenir la rétrocompatibilité

Lors de la décomposition de la base de données, le maintien de la rétrocompatibilité est essentiel pour des transitions fluides. Maintenez les procédures de base de données existantes temporairement en place tout en implémentant progressivement de nouvelles fonctionnalités. Utilisez le contrôle de version pour suivre toutes les modifications et gérer simultanément plusieurs versions de base de données. Prévoyez une période de coexistence prolongée pendant laquelle les systèmes source et cible doivent fonctionner de manière fiable. Cela laisse le temps de tester et de valider le nouveau système avant de retirer les composants existants. Cette approche minimise les interruptions d'activité et fournit un filet de sécurité permettant de revenir en arrière si nécessaire.

Plan de réduction d'urgence

Une stratégie de restauration complète est essentielle pour une décomposition sûre des bases de données. Implémentez des indicateurs de fonctionnalité dans votre code pour contrôler quelle version de la logique métier est active. Cela vous permet de basculer instantanément entre les nouvelles implémentations et les implémentations d'origine sans modifier le déploiement. Cette approche permet un contrôle précis de la transition et vous aide à revenir en arrière rapidement en cas de problème. Conservez la logique d'origine sous forme de sauvegarde vérifiée et maintenez des procédures de restauration détaillées qui spécifient les déclencheurs, les responsabilités et les étapes de restauration.

Testez régulièrement ces scénarios de réduction dans différentes conditions pour valider leur efficacité et assurez-vous que les équipes connaissent bien les procédures d'urgence. Les indicateurs de fonctionnalité permettent également des déploiements progressifs en activant de manière sélective de nouvelles fonctionnalités pour des groupes d'utilisateurs ou des transactions spécifiques. Cela fournit un niveau supplémentaire d'atténuation des risques pendant la transition.

Découplage des relations entre les tables lors de la décomposition de la base de données

Cette section fournit des conseils sur la décomposition des relations complexes entre les tables et les opérations JOIN lors de la décomposition d'une base de données monolithique. Une jointure de table combine les lignes de deux tables ou plus en fonction d'une colonne associée entre elles. L'objectif de la séparation de ces relations est de réduire le couplage élevé entre les tables tout en préservant l'intégrité des données dans les microservices.

Cette section contient les rubriques suivantes :

- Stratégie de dénormalisation
- Reference-by-key stratégie
- Motif CQRS
- Synchronisation des données basée sur les événements
- Implémentation d'alternatives aux jointures de tables
- Exemple basé sur un scénario

Stratégie de dénormalisation

La dénormalisation est une stratégie de conception de base de données qui consiste à introduire intentionnellement de la redondance en combinant ou en dupliquant des données entre des tables. Lorsque vous divisez une grande base de données en petites bases de données, il peut être judicieux de dupliquer certaines données entre les services. Par exemple, le stockage des informations de base sur les clients, telles que le nom et les adresses e-mail, à la fois dans un service marketing et dans un service de commande élimine le besoin de recherches interservices constantes. Le service marketing peut avoir besoin des préférences et des coordonnées des clients pour le ciblage des campagnes, tandis que le service des commandes a besoin des mêmes données pour le traitement des commandes et les notifications. Bien que cela crée une certaine redondance des données, cela peut améliorer considérablement les performances et l'indépendance du service, permettant à l'équipe marketing de gérer ses campagnes sans dépendre des recherches en temps réel du service client.

Lorsque vous mettez en œuvre la dénormalisation, concentrez-vous sur les champs fréquemment consultés que vous identifiez grâce à une analyse minutieuse des modèles

d'accès aux données. Vous pouvez utiliser des outils, tels que Oracle AWR des rapports ou des rapportspg_stat_statements, pour comprendre quelles données sont généralement extraites ensemble. Les experts du domaine peuvent également fournir des informations précieuses sur les groupements de données naturelles. N'oubliez pas que la dénormalisation n'est pas une all-or-nothing approche, mais uniquement des données dupliquées qui améliorent de manière démontrable les performances du système ou réduisent les dépendances complexes.

Reference-by-key stratégie

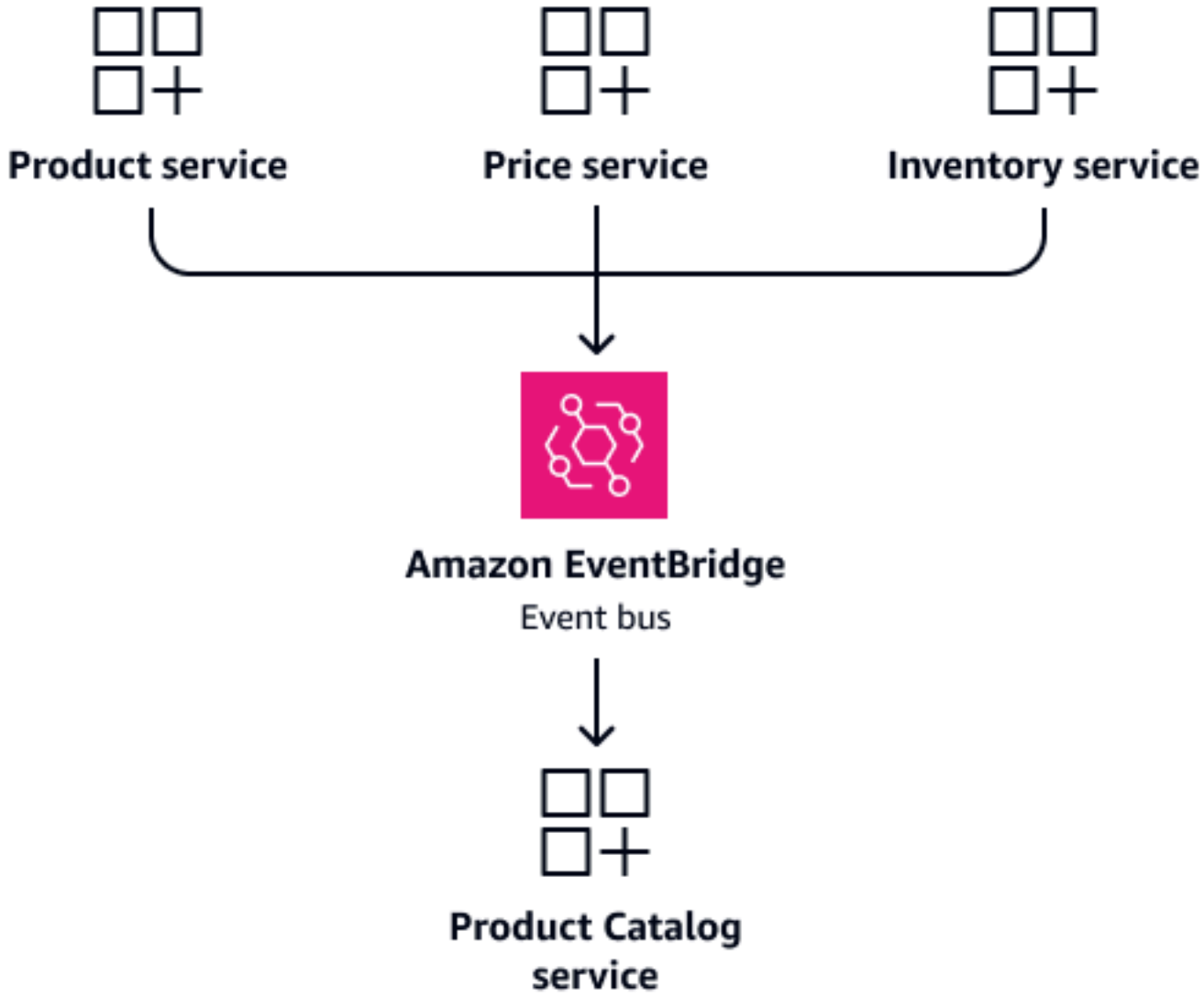
Une reference-by-key stratégie est un modèle de conception de base de données dans lequel les relations entre les entités sont maintenues par le biais de clés uniques plutôt que de stocker les données associées réelles. Au lieu des relations traditionnelles à clé étrangère, les microservices modernes stockent souvent uniquement les identifiants uniques des données associées. Par exemple, plutôt que de conserver toutes les informations du client dans le tableau des commandes, le service des commandes enregistre uniquement l'identifiant du client et récupère des informations supplémentaires sur le client via un appel d'API en cas de besoin. Cette approche préserve l'indépendance des services tout en garantissant l'accès aux données associées.

Motif CQRS

Le modèle CQRS (Command Query Responsibility Segrégation) sépare les opérations de lecture et d'écriture d'un magasin de données. Ce modèle est particulièrement utile dans les systèmes complexes nécessitant de hautes performances, en particulier ceux soumis à des read/write charges asymétriques. Si votre application a fréquemment besoin de combiner des données provenant de plusieurs sources, vous pouvez créer un modèle CQRS dédié au lieu de jointures complexes. Par exemple, plutôt que de joindre `Product` des `Inventory` tables à chaque demande, maintenez une `Product Catalog` table consolidée contenant les données nécessaires. `Pricing` Les avantages de cette approche peuvent l'emporter sur les coûts de la table supplémentaire.

Envisagez un scénario dans lequel `ProductPrice`, et les `Inventory` services ont fréquemment besoin d'informations sur les produits. Au lieu de configurer ces services pour accéder directement aux tables partagées, créez un `Product Catalog` service dédié. Ce service gère sa propre base de données qui contient les informations consolidées sur les produits. Il agit comme une source unique de vérité pour les requêtes relatives aux produits. Lorsque les détails du produit, les prix ou les niveaux de stock changent, les services concernés peuvent publier des événements pour mettre à jour le `Product Catalog` service. Cela garantit la cohérence des données tout en préservant

l'indépendance des services. L'image suivante montre cette configuration, où Amazon EventBridge fait office de bus d'événements.



Comme indiqué dans la section suivanteSynchronisation des données basée sur les événements, maintenez le modèle CQRS à jour au fil des événements. Lorsque les détails des produits, les prix ou les niveaux de stock changent, les services concernés publient des événements. Le `Product Catalog` service s'abonne à ces événements et met à jour sa vue consolidée. Cela permet des lectures rapides sans jointure complexe et préserve l'indépendance du service.

Synchronisation des données basée sur les événements

La synchronisation des données basée sur les événements est un modèle dans lequel les modifications apportées aux données sont capturées et propagées sous forme d'événements, ce

qui permet à différents systèmes ou composants de maintenir des états de données synchronisés. Lorsque les données changent, au lieu de mettre à jour immédiatement toutes les bases de données associées, publiez un événement pour informer les services abonnés. Par exemple, lorsqu'un client change d'adresse de livraison dans le `Customer` service, un `CustomerUpdated` événement déclenche des mises à jour du `Order` service et du `Delivery` service selon le calendrier de chaque service. Cette approche remplace les jointures de table rigides par des mises à jour flexibles et évolutives basées sur les événements. Certains services peuvent brièvement contenir des données périmées, mais le compromis réside dans l'amélioration de l'évolutivité du système et de l'indépendance des services.

Implémentation d'alternatives aux jointures de tables

Commencez la décomposition de votre base de données par des opérations de lecture, car elles sont généralement plus simples à migrer et à valider. Une fois les chemins de lecture stables, abordez les opérations d'écriture les plus complexes. Pour les exigences critiques et de haute performance, envisagez d'implémenter le modèle CQRS. Utilisez une base de données distincte et optimisée pour les lectures tout en conservant une autre pour les écritures.

Créez des systèmes résilients en ajoutant une logique de nouvelle tentative pour les appels interservices et en implémentant les couches de mise en cache appropriées. Surveillez de près les interactions entre les services et configurez des alertes pour les problèmes de cohérence des données. L'objectif final n'est pas la cohérence parfaite partout, mais la création de services indépendants performants tout en maintenant une précision des données acceptable pour les besoins de votre entreprise.

La nature découplée des microservices introduit les nouvelles complexités suivantes en matière de gestion des données :

- Les données sont distribuées. Les données se trouvent désormais dans des bases de données distinctes, gérées par des services indépendants.
- La synchronisation en temps réel entre les services est souvent peu pratique, ce qui nécessite un éventuel modèle de cohérence.
- Les opérations qui se produisaient auparavant dans le cadre d'une seule transaction de base de données concernent désormais plusieurs services.

Pour relever ces défis, procédez comme suit :

- Mettez en œuvre une architecture axée sur les événements : utilisez les files d'attente de messages et la publication d'événements pour propager les modifications de données entre les services. Pour plus d'informations, voir Création d'architectures pilotées par les événements sur un terrain sans serveur.
- Adoptez le modèle d'orchestration Saga : ce modèle vous aide à gérer les transactions distribuées et à préserver l'intégrité des données sur l'ensemble des services. Pour plus d'informations, voir Création d'une application distribuée sans serveur à l'aide d'un modèle d'orchestration Saga sur AWS Blogs.
- Conception en cas d'échec : intégrez des mécanismes de nouvelle tentative, des disjoncteurs et des transactions de compensation pour gérer les problèmes de réseau ou les pannes de service.
- Utiliser l'estampillage de version : suivez les versions des données pour gérer les conflits et vous assurer que les mises à jour les plus récentes sont appliquées.
- Réconciliation régulière — Mettez en œuvre des processus de synchronisation des données périodiques pour détecter et corriger les éventuelles incohérences.

Exemple basé sur un scénario

L'exemple de schéma suivant comporte deux tables, une `Customer` table et une `Order` table :

```
-- 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)
```

```
);
```

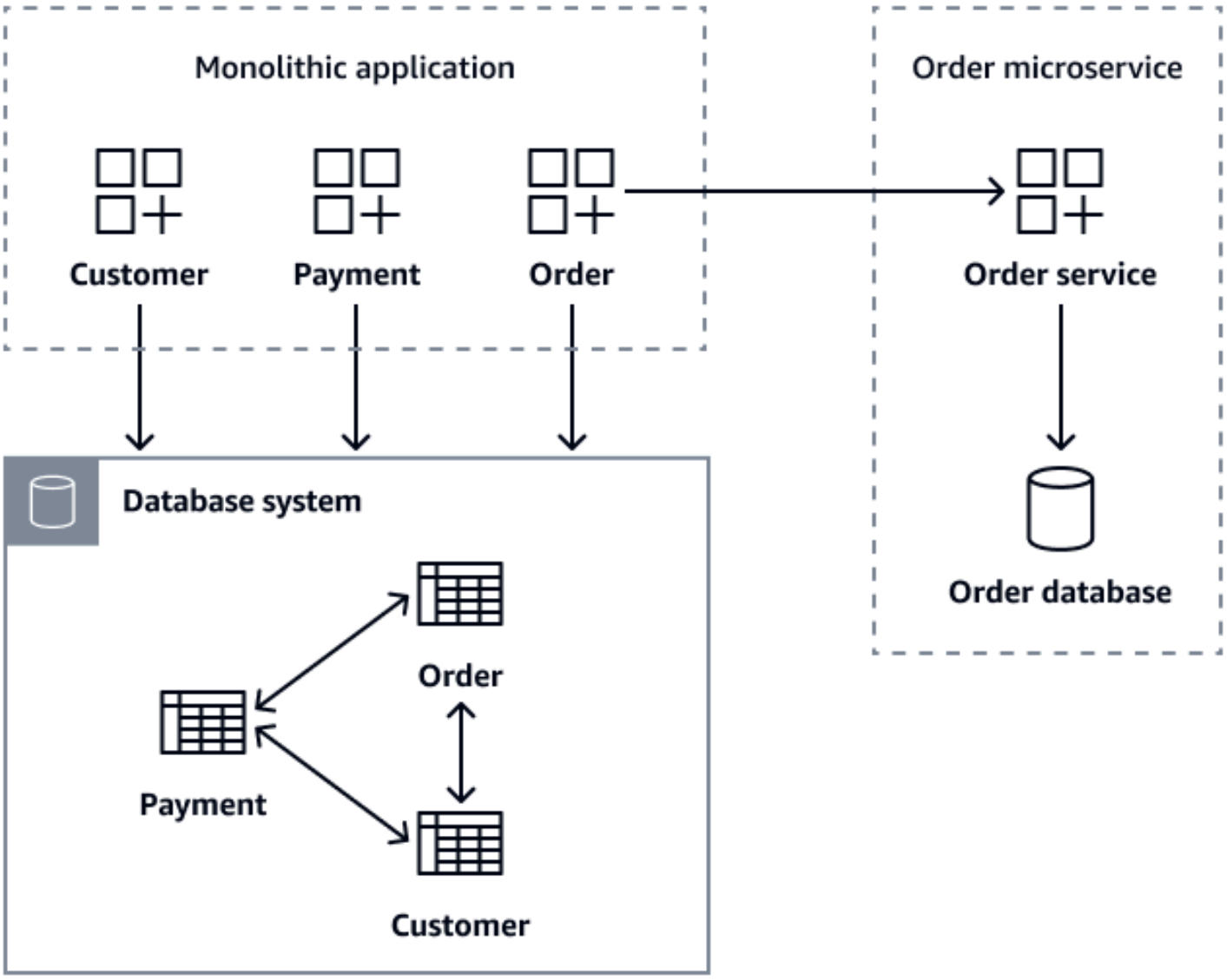
Voici un exemple de la façon dont vous pourriez utiliser une approche dénormalisée :

```
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)
);
```

Le nom du client et les adresses e-mail du nouveau `Order` tableau sont dénormalisés. Le `customer_id` est référencé et il n'existe aucune contrainte de clé étrangère avec la `Customer` table. Les avantages de cette approche dénormalisée sont les suivants :

- Le `Order` service peut afficher l'historique des commandes avec les détails du client, et il ne nécessite pas d'appels d'API au `Customer` microservice.
- Si le `Customer` service est en panne, il reste pleinement fonctionnel. `Order`
- Les requêtes relatives au traitement des commandes et à la création de rapports sont plus rapides.

Le schéma suivant montre une application monolithique qui récupère
les données de commande à l'aide de`getOrder(customer_id)`,
`getOrder(order_id)getCustomerOders(customer_id)`, et d'appels `createOrder(Order order)` d'API au `Order` microservice.



Pendant la migration des microservices, vous pouvez conserver la `Order` table dans la base de données monolithique à titre de mesure de sécurité transitoire, afin de garantir que l'ancienne application reste fonctionnelle. Cependant, il est essentiel que toutes les nouvelles opérations liées aux commandes soient acheminées via l'API de `Order` microservice, qui gère sa propre base de données tout en écrivant simultanément dans l'ancienne base de données en tant que sauvegarde. Ce modèle à double écriture constitue un filet de sécurité. Il permet une migration progressive tout en préservant la stabilité du système. Une fois que tous les clients ont migré avec succès vers le nouveau microservice, vous pouvez désactiver l'ancienne `Order` table dans la base de données monolithique. Après avoir décomposé l'application monolithique et sa base de données en `Order` microservices distincts`Customer`, le principal défi consiste à maintenir la cohérence des données.

Bonnes pratiques pour la décomposition des bases de données

Lors de la décomposition d'une base de données monolithique, les organisations doivent établir des cadres clairs pour suivre les progrès, maintenir la connaissance du système et relever les défis émergents. Cette section fournit les meilleures pratiques pour mesurer le succès de la décomposition, gérer la documentation essentielle, mettre en œuvre des processus d'amélioration continue et relever les défis courants. La compréhension et le respect de ces directives vous permettent de vous assurer que les efforts de décomposition des bases de données produisent les avantages escomptés tout en minimisant les perturbations opérationnelles et les dettes techniques.

Cette section contient les rubriques suivantes :

- Mesurer le succès
- Exigences en matière de documentation
- Stratégie d'amélioration continue
- Surmonter les défis courants liés à la décomposition des bases de données

Mesurer le succès

Suivez le succès de la décomposition grâce à une combinaison de mesures techniques, opérationnelles et commerciales. Sur le plan technique, surveillez les temps de réponse aux requêtes, les améliorations de la disponibilité du système et l'augmentation de la fréquence de déploiement. Sur le plan opérationnel, mesurez la réduction des incidents, la vitesse de résolution des problèmes et les améliorations de l'utilisation des ressources. Pour le développement, suivez la vitesse de mise en œuvre des fonctionnalités, l'accélération du cycle de publication et la réduction des dépendances entre les équipes. Les impacts commerciaux devraient se traduire par une réduction des coûts d'exploitation, une accélération time-to-market et une amélioration de la satisfaction des clients. Ces métriques sont souvent définies au cours de la phase de cadrage. Pour plus d'informations, consultez la section Définition de l'étendue et des exigences relatives à la décomposition des bases de données dans ce guide.

Exigences en matière de documentation

Maintenez la documentation de l'architecture up-to-date du système avec des limites de service, des flux de données et des spécifications d'interface clairs. Utilisez les enregistrements de décisions d'architecture (ADRs) pour saisir les décisions techniques clés, y compris leur contexte, leurs conséquences et les alternatives envisagées. Par exemple, expliquez pourquoi des services spécifiques ont d'abord été séparés ou comment certains compromis ont été faits en matière de cohérence des données.

Planifiez des révisions mensuelles de l'architecture pour évaluer l'état du système grâce à des indicateurs clés : tendances en matière de performances, conformité en matière de sécurité et dépendances entre services. Incluez les commentaires des équipes de développement sur les défis d'intégration et les problèmes opérationnels. Ce cycle de révision régulier vous aide à identifier rapidement les problèmes émergents et à vérifier que les efforts de décomposition restent conformes aux objectifs commerciaux.

Stratégie d'amélioration continue

Traitez la décomposition de la base de données comme un processus itératif et non comme un projet ponctuel. Surveillez les indicateurs de performance du système et les interactions entre les services pour identifier les opportunités d'optimisation. Chaque trimestre, priorisez le traitement de la dette technique en fonction de l'impact opérationnel et des coûts de maintenance. Par exemple, automatisez les opérations de base de données fréquemment effectuées, améliorez la couverture de surveillance et affinez les procédures de déploiement en fonction des modèles appris.

Surmonter les défis courants liés à la décomposition des bases de données

L'optimisation des performances nécessite une approche multidimensionnelle. Mettez en œuvre une mise en cache stratégique aux limites des services, optimisez les modèles de requêtes en fonction de l'utilisation réelle et surveillez en permanence les indicateurs clés. Résolvez les problèmes de performance de manière proactive en analysant les tendances et en définissant des seuils d'intervention clairs.

Les défis liés à la cohérence des données exigent des choix architecturaux judicieux. Implémentez des modèles pilotés par les événements pour les mises à jour interservices et utilisez des modèles d'orchestration de saga pour les transactions complexes. Définissez des limites de service claires et

acceptez une éventuelle cohérence lorsque les exigences commerciales le permettent. Cet équilibre entre cohérence et autonomie du service est crucial pour une décomposition réussie.

L'excellence opérationnelle nécessite l'automatisation des tâches de routine et des procédures standardisées pour tous les services. Maintenez une surveillance complète avec des seuils d'alerte clairs et investissez dans la formation régulière des équipes aux nouveaux modèles et outils. Cette approche systématique des opérations favorise la fiabilité de la prestation de services tout en gérant la complexité.

FAQ sur la décomposition des bases de données

Cette section FAQ complète aborde les questions et les défis les plus courants auxquels les entreprises sont confrontées lorsqu'elles entreprennent des projets de décomposition de bases de données. De la définition de la portée et des exigences initiales à la migration des procédures stockées, ces questions fournissent des informations pratiques et des approches stratégiques pour aider les équipes à mener à bien leur parcours de modernisation des bases de données. Que vous soyez en phase de planification ou que vous exécutiez déjà votre stratégie de décomposition, ces réponses peuvent vous aider à éviter les pièges courants et à mettre en œuvre les meilleures pratiques pour obtenir des résultats optimaux.

Cette section contient les rubriques suivantes :

- FAQs sur la définition du champ d'application et des exigences
- FAQs à propos du contrôle de l'accès aux bases
- FAQs à propos de l'analyse de la cohésion et du couplage
- FAQs à propos de la migration de la logique métier vers la couche applicative

FAQs sur la définition du champ d'application et des exigences

La Définition de la portée et des exigences relatives à la décomposition des bases de données section de ce guide explique comment analyser les interactions, cartographier les dépendances et établir des critères de réussite. Cette section FAQ aborde les principales questions relatives à l'établissement et à la gestion des limites des projets. Que vous soyez confronté à des contraintes techniques floues, à des besoins ministériels contradictoires ou à l'évolution des exigences opérationnelles, ces documents FAQs fournissent des conseils pratiques sur le maintien d'une approche équilibrée.

Cette section contient les questions suivantes :

- Dans quelle mesure la définition initiale du champ d'application doit-elle être détaillée ?
- Et si je découvre des dépendances supplémentaires après avoir démarré le projet ?
- Comment gérer les parties prenantes de différents départements qui ont des exigences contradictoires ?
- Quel est le meilleur moyen d'évaluer les contraintes techniques lorsque la documentation est insuffisante ou obsolète ?

- Comment trouver un équilibre entre les besoins commerciaux immédiats et les objectifs techniques à long terme ?
- Comment m'assurer que je ne passe pas à côté des exigences critiques de la part de parties prenantes silencieuses ?
- Ces recommandations s'appliquent-elles aux bases de données mainframe monolithiques ?

Dans quelle mesure la définition initiale du champ d'application doit-elle être détaillée ?

En partant des besoins de vos clients, définissez la portée du projet avec suffisamment de détails pour identifier les limites du système et les dépendances critiques, tout en préservant la flexibilité nécessaire à la découverte. Cartographiez les éléments essentiels, notamment les interfaces du système, les principales parties prenantes et les principales contraintes techniques. Commencez modestement en sélectionnant une partie limitée et à faible risque du système qui fournit une valeur mesurable. Cette approche aide les équipes à apprendre et à ajuster leurs stratégies avant de s'attaquer à des composants plus complexes.

Documentez les exigences commerciales critiques qui sous-tendent l'effort de décomposition, mais évitez de trop spécifier les détails susceptibles de changer au cours de la mise en œuvre. Cette approche équilibrée permet aux équipes d'avancer avec clarté tout en restant adaptables aux nouvelles connaissances et aux nouveaux défis qui émergent au cours du processus de modernisation.

Et si je découvre des dépendances supplémentaires après avoir démarré le projet ?

Attendez-vous à découvrir des dépendances supplémentaires au fur et à mesure de l'avancement du projet. Tenez un journal des dépendances en temps réel et effectuez des examens réguliers du périmètre pour évaluer l'impact sur les délais et les ressources. Mettez en œuvre un processus clair de gestion du changement et incluez une période tampon dans les plans de projet pour faire face aux découvertes inattendues. L'objectif n'est pas d'empêcher les changements mais de les gérer efficacement. Cela permet aux équipes de s'adapter rapidement tout en maintenant la dynamique du projet.

Comment gérer les parties prenantes de différents départements qui ont des exigences contradictoires ?

Gérez les exigences ministérielles contradictoires grâce à une hiérarchisation claire basée sur la valeur commerciale et l'impact sur le système. Obtenez le parrainage de la direction pour prendre les décisions clés et résoudre rapidement les conflits. Planifiez des réunions régulières d'alignement des parties prenantes pour discuter des compromis et maintenir la transparence. Documentez toutes les décisions et leur justification afin de promouvoir une communication claire et de maintenir la dynamique du projet. Concentrez les discussions sur les avantages commerciaux quantifiables plutôt que sur les préférences ministérielles.

Quel est le meilleur moyen d'évaluer les contraintes techniques lorsque la documentation est insuffisante ou obsolète ?

Lorsque vous êtes confronté à une mauvaise documentation, associez l'analyse traditionnelle aux outils d'intelligence artificielle modernes. Utilisez de grands modèles de langage (LLMs) pour analyser les référentiels de code, les journaux et la documentation existante afin d'identifier les modèles et les contraintes potentielles. Interrogez des développeurs et des architectes de bases de données expérimentés pour valider les résultats de l'IA et découvrir des contraintes non documentées. Déployez des outils de surveillance dotés de capacités d'IA améliorées afin d'observer le comportement du système et de prévoir les problèmes potentiels.

Créez de petites expériences techniques qui valident vos hypothèses. Vous pouvez utiliser des outils de test basés sur l'IA pour accélérer le processus. Documentez les résultats dans une base de connaissances qui peut être continuellement améliorée grâce à des mises à jour assistées par l'IA. Envisagez de faire appel à des experts en la matière pour des domaines complexes et utilisez des outils de programmation par paire basés sur l'IA pour accélérer leurs efforts d'analyse et de documentation.

Comment trouver un équilibre entre les besoins commerciaux immédiats et les objectifs techniques à long terme ?

Créez une feuille de route de projet progressive qui aligne les besoins commerciaux immédiats avec les objectifs techniques à long terme. Identifiez rapidement les gains rapides qui apportent une valeur tangible afin de renforcer la confiance des parties prenantes. Décomposez la décomposition en étapes claires. Chacun devrait apporter des avantages commerciaux mesurables tout en progressant

vers les objectifs architecturaux. Maintenez la flexibilité nécessaire pour répondre aux besoins commerciaux urgents grâce à des révisions et des ajustements réguliers de la feuille de route.

Comment m'assurer que je ne passe pas à côté des exigences critiques de la part de parties prenantes silencieuses ?

Cartographiez toutes les parties prenantes potentielles au sein de l'organisation, y compris les propriétaires de systèmes en aval et les utilisateurs indirects. Créez plusieurs canaux de feedback par le biais d'entretiens structurés, d'ateliers et de sessions de révision régulières. Construisez proof-of-concepts et prototypez pour rendre les exigences tangibles et susciter des discussions constructives. Par exemple, un simple tableau de bord qui montre les dépendances du système révèle souvent des parties prenantes cachées et des exigences qui n'étaient pas apparentes au départ.

Organisez des sessions de validation régulières avec les parties prenantes, qu'elles soient bruyantes ou discrètes, et assurez-vous que tous les points de vue sont pris en compte. Les informations critiques proviennent souvent des personnes les plus proches des opérations quotidiennes plutôt que des voix les plus fortes lors des réunions de planification.

Ces recommandations s'appliquent-elles aux bases de données mainframe monolithiques ?

La méthodologie décrite dans ce guide s'applique également à la décomposition de bases de données mainframe monolithiques. Les principaux défis liés à ces bases de données sont la gestion des exigences des différentes parties prenantes. Les recommandations technologiques de ce guide peuvent s'appliquer aux bases de données mainframe monolithiques. Si le mainframe possède une base de données relationnelle, telle qu'une base de données de traitement des transactions en ligne (OLTP), de nombreuses recommandations s'appliquent. Pour les bases de données de traitement analytique en ligne (OLAP), telles que celles utilisées pour générer des rapports commerciaux, seules certaines recommandations s'appliquent.

FAQs à propos du contrôle de l'accès aux bases

Le contrôle de l'accès à la base de données à l'aide du modèle de service d'encapsulation de base de données est décrit dans la Contrôle de l'accès aux bases de données pendant la décomposition section de ce guide. Cette section FAQ aborde les préoccupations et questions courantes concernant l'introduction d'un service d'encapsulation de base de données, notamment son impact potentiel

sur les performances, la gestion des procédures stockées existantes, la gestion des transactions complexes et la supervision des modifications de schéma.

Cette section contient les questions suivantes :

- Le service d'emballage ne deviendra-t-il pas un nouveau goulot d'étranglement ?
- Qu'advient-il des procédures stockées existantes ?
- Comment gérer les modifications du schéma pendant la transition ?

Le service d'emballage ne deviendra-t-il pas un nouveau goulot d'étranglement ?

Bien que le service d'encapsulation de base de données ajoute un saut réseau supplémentaire, l'impact est généralement minime. Vous pouvez faire évoluer le service horizontalement, et les avantages d'un accès contrôlé l'emportent généralement sur le faible coût des performances. Considérez cela comme un compromis temporaire entre performance et maintenabilité.

Qu'advient-il des procédures stockées existantes ?

Dans un premier temps, le service d'encapsulation de base de données peut exposer les procédures stockées en tant que méthodes de service. Au fil du temps, vous pouvez progressivement transférer la logique dans la couche d'application, ce qui améliore les tests et le contrôle des versions. Migrez la logique métier de manière incrémentielle afin de minimiser les risques.

Comment gérer les modifications du schéma pendant la transition ?

Centralisez le contrôle des modifications de schéma par le biais de l'équipe du service Wrapper. Cette équipe est chargée de maintenir une visibilité complète auprès de tous les consommateurs. Cette équipe examine les modifications proposées pour en déterminer l'impact à l'échelle du système, assure la coordination avec les équipes concernées et met en œuvre les modifications en utilisant un processus de déploiement contrôlé. Par exemple, lors de l'ajout de nouveaux champs, cette équipe doit maintenir la rétrocompatibilité en implémentant des valeurs par défaut ou en autorisant initialement les valeurs nulles.

Établissez un processus clair de gestion des modifications qui inclut l'évaluation de l'impact, les exigences de test et les procédures d'annulation. Utilisez des outils de gestion des versions de base de données et conservez une documentation claire de toutes les modifications. Cette approche

centralisée empêche les modifications du schéma de perturber les services dépendants et préserve la stabilité du système.

FAQs à propos de l'analyse de la cohésion et du couplage

Comprendre et analyser efficacement le couplage et la cohésion des bases de données est essentiel à la réussite de la décomposition des bases de données. Le couplage et la cohésion sont abordés dans la Analyse de la cohésion et du couplage pour la décomposition des bases de données section de ce guide. Cette section FAQ aborde les principales questions relatives à l'identification des niveaux de granularité appropriés, à la sélection des bons outils d'analyse, à la documentation des résultats et à la hiérarchisation des problèmes de couplage.

Cette section contient les questions suivantes :

- Comment identifier le bon niveau de granularité lors de l'analyse du couplage ?
- Quels outils puis-je utiliser pour analyser le couplage et la cohésion des bases de données ?
- Quelle est la meilleure façon de documenter les résultats de couplage et de cohésion ?
- Comment puis-je hiérarchiser les problèmes de couplage à résoudre en premier ?
- Comment gérer les transactions qui s'étendent sur plusieurs opérations ?

Comment identifier le bon niveau de granularité lors de l'analyse du couplage ?

Commencez par une analyse globale des relations entre les bases de données, puis approfondissez systématiquement pour identifier les points de séparation naturels. Utilisez les outils d'analyse de base de données pour cartographier les relations au niveau des tables, les dépendances des schémas et les limites des transactions. Par exemple, examinez les modèles de jointure dans les requêtes SQL pour comprendre les dépendances d'accès aux données. Vous pouvez également analyser les journaux de transactions pour identifier les limites des processus métier.

Concentrez-vous sur les domaines où le couplage est naturellement minimal. Ils correspondent souvent aux limites du domaine d'activité et représentent des points de décomposition optimaux. Lorsque vous déterminez les limites de service appropriées, tenez compte à la fois du couplage technique (tel que les tables partagées et les clés étrangères) et du couplage commercial (tel que les flux de processus et les besoins en matière de reporting).

Quels outils puis-je utiliser pour analyser le couplage et la cohésion des bases de données ?

Vous pouvez utiliser une combinaison d'outils automatisés et d'analyses manuelles pour évaluer le couplage et la cohésion des bases de données. Les outils suivants peuvent vous aider dans cette évaluation :

- Outils de visualisation de schémas : vous pouvez utiliser des outils tels que SchemaSpyou pgAdminpour générer des diagrammes ER. Ces diagrammes révèlent les relations entre les tables et les points de couplage potentiels.
- Outils d'analyse des requêtes : vous pouvez utiliser pg_stat_statementsou SQL Server Query Storepour identifier les tables fréquemment jointes et les modèles d'accès.
- Outils de profilage de base de données : outils tels que Oracle SQL Developerou MySQL Workbenchfournissant des informations sur les performances des requêtes et les dépendances des données.
- Outils de mappage des dépendances : le AWS Schema Conversion Tool (AWS SCT) peut vous aider à visualiser les relations entre les schémas et à identifier les composants étroitement couplés. vFunctionpeut vous aider à identifier les limites du domaine en analysant les limites fonctionnelles et de domaine de l'application.
- Outils de surveillance des transactions : vous pouvez utiliser des outils spécifiques à la base de données, tels que Oracle Enterprise Managerou SQL Server Extended Events, pour analyser les limites des transactions.
- Outils de migration de logique métier : vous pouvez utiliser Ispirerou générer des outils d'IA, tels qu'Amazon Q Developer Kiro, ou pour convertir la logique métier de base de données pour la couche application, telle que la conversion en Java.

Combinez ces analyses automatisées avec un examen manuel des processus métier et des connaissances du domaine pour bien comprendre le couplage des systèmes. Cette approche multidimensionnelle garantit que les perspectives techniques et commerciales sont prises en compte dans votre stratégie de décomposition.

Quelle est la meilleure façon de documenter les résultats de couplage et de cohésion ?

Créez une documentation complète qui visualise les relations entre les bases de données et les modèles d'utilisation. Les types de ressources que vous pouvez utiliser pour enregistrer vos résultats sont les suivants :

- Matrices de dépendances : cartographiez les dépendances des tables et mettez en évidence les zones à fort couplage.
- Diagrammes de relations : utilisez les diagrammes ER pour montrer les connexions entre les schémas et les relations entre clés étrangères.
- Cartes thermiques d'utilisation des tables : visualisez la fréquence des requêtes et les modèles d'accès aux données entre les tables.
- Diagrammes de flux de transactions — Documentez les transactions multi-tables et leurs limites.
- Cartes des limites des domaines — Décrivez les limites de service potentielles en fonction des domaines commerciaux.

Combinez ces artefacts dans un document et mettez-le régulièrement à jour au fur et à mesure que la décomposition progresse. Pour les diagrammes, vous pouvez utiliser des outils tels que draw.ioou Lucidchart. Envisagez de mettre en place un wiki pour faciliter l'accès et la collaboration des équipes. Cette approche documentaire à multiples facettes fournit une compréhension claire et partagée du couplage et de la cohésion des systèmes.

Comment puis-je hiérarchiser les problèmes de couplage à résoudre en premier ?

Priorisez les problèmes de couplage sur la base d'une évaluation équilibrée des facteurs commerciaux et techniques. Évaluez chaque problème en fonction de l'impact commercial (comme le chiffre d'affaires et l'expérience client), des risques techniques (tels que la stabilité du système et l'intégrité des données), des efforts de mise en œuvre et des capacités de l'équipe. Créez une matrice de priorisation qui note chaque problème de 1 à 5 selon ces dimensions. Cette matrice vous aide à identifier les opportunités les plus intéressantes présentant des risques gérables.

Commencez par des changements à fort impact et à faible risque qui s'alignent sur l'expertise de l'équipe existante. Cela vous aide à renforcer la confiance organisationnelle et l'élan nécessaires pour des changements plus complexes. Cette approche favorise une exécution réaliste et maximise la

valeur commerciale. Passez régulièrement en revue et ajustez les priorités pour vous aider à rester en phase avec l'évolution des besoins de l'entreprise et des capacités de l'équipe.

Comment gérer les transactions qui s'étendent sur plusieurs opérations ?

Gérez les transactions multi-opérations grâce à une coordination des niveaux de service soigneusement conçue. Implémentez des modèles de saga pour les transactions distribuées complexes. Divisez-les en étapes plus petites et réversibles qui peuvent être gérées indépendamment. Par exemple, un flux de traitement des commandes peut être divisé en étapes distinctes pour la vérification des stocks, le traitement des paiements et la création de commandes, chacune ayant son propre mécanisme de compensation.

Dans la mesure du possible, redéfinissez les opérations pour qu'elles soient plus atomiques, ce qui réduit le besoin de transactions distribuées. Lorsque les transactions distribuées sont inévitables, mettez en œuvre des mécanismes de suivi et de compensation robustes pour promouvoir la cohérence des données. Surveillez les taux d'achèvement des transactions et mettez en œuvre des procédures claires de correction des erreurs afin de maintenir la fiabilité du système.

FAQs à propos de la migration de la logique métier vers la couche applicative

La migration de la logique métier de la base de données vers la couche application est un aspect critique et complexe de la modernisation des bases de données. Cette migration de logique métier est abordée dans la Migration de la logique métier de la base de données vers la couche application section de ce guide. Cette section FAQ aborde les questions courantes relatives à la gestion efficace de cette transition, qu'il s'agisse de sélectionner les candidats initiaux pour la migration ou de gérer des procédures stockées et des déclencheurs complexes.

Cette section contient les questions suivantes :

- Comment identifier les procédures stockées à migrer en premier ?
- Quels sont les risques liés au transfert de la logique vers la couche application ?
- Comment maintenir les performances lorsque je déplace la logique hors de la base de données ?
- Que dois-je faire avec les procédures stockées complexes impliquant plusieurs tables ?
- Comment gérer les déclencheurs de base de données lors de la migration ?
- Quel est le meilleur moyen de tester la logique métier migrée ?

- Comment gérer la période de transition lorsque la logique de base de données et d'application existe à la fois ?
- Comment gérer les scénarios d'erreur dans la couche d'application qui étaient auparavant gérés par la base de données ?

Comment identifier les procédures stockées à migrer en premier ?

Commencez par identifier les procédures stockées qui offrent la meilleure combinaison entre un faible risque et une valeur d'apprentissage élevée. Concentrez-vous sur des procédures présentant un minimum de dépendances, des fonctionnalités claires et un impact commercial non critique. Ils constituent des candidats idéaux pour la migration initiale, car ils aident l'équipe à renforcer la confiance et à établir des modèles. Par exemple, choisissez des procédures qui gèrent des opérations de données simples plutôt que celles qui gèrent des transactions complexes ou une logique métier critique.

Utilisez des outils de surveillance de base de données pour analyser les modèles d'utilisation et identifier les procédures rarement consultées en tant que premiers candidats. Cette approche minimise les risques commerciaux tout en fournissant une expérience précieuse pour aborder ultérieurement des migrations plus complexes. Évaluez chaque procédure en fonction de sa complexité, de son caractère critique pour l'entreprise et de ses niveaux de dépendance afin de créer une séquence de migration hiérarchisée.

Quels sont les risques liés au transfert de la logique vers la couche application ?

Le déplacement de la logique de base de données vers la couche application présente plusieurs défis majeurs. Les performances du système peuvent se dégrader en raison de l'augmentation des appels réseau, en particulier pour les opérations gourmandes en données qui étaient auparavant gérées au sein de la base de données. La gestion des transactions devient de plus en plus complexe et nécessite une coordination minutieuse afin de préserver l'intégrité des données dans les opérations distribuées. Garantir la cohérence des données devient un défi, en particulier pour les opérations qui reposaient auparavant sur des contraintes au niveau de la base de données.

Les perturbations commerciales potentielles pendant la migration et la courbe d'apprentissage des développeurs constituent également des préoccupations majeures. Limitez ces risques grâce à une planification minutieuse, à des tests approfondis dans des environnements par étapes et à une migration progressive qui commence par les composants moins critiques. Mettez en œuvre

des procédures de surveillance et d'annulation robustes pour identifier et résoudre rapidement les problèmes de production.

Comment maintenir les performances lorsque je déplace la logique hors de la base de données ?

Mettez en œuvre des mécanismes de mise en cache appropriés pour les données fréquemment consultées, optimisez les modèles d'accès aux données afin de minimiser les appels réseau et utilisez le traitement par lots pour les opérations en masse. Pour les non-time-critical opérations, envisagez le traitement asynchrone afin d'améliorer la réactivité du système.

Surveillez de près les indicateurs de performance des applications et ajustez-les selon les besoins. Par exemple, vous pouvez remplacer plusieurs opérations sur une seule ligne par un traitement en bloc, vous pouvez mettre en cache des données de référence qui changent rarement et vous pouvez optimiser les modèles de requêtes pour réduire le transfert de données. Des tests et des réglages réguliers des performances aident le système à maintenir des temps de réponse acceptables et améliorent la maintenabilité et l'évolutivité.

Que dois-je faire avec les procédures stockées complexes impliquant plusieurs tables ?

Approchez des procédures stockées complexes comportant plusieurs tables par le biais d'une décomposition systématique. Commencez par les diviser en composants plus petits et logiquement cohérents, puis identifiez clairement les limites des transactions et les dépendances entre les données. Créez des interfaces de service pour chaque composant logique. Cela vous permet de migrer progressivement sans perturber les fonctionnalités existantes.

Mettez en œuvre une step-by-step migration, en commençant par les composants les moins couplés. Pour les procédures très complexes, pensez à les conserver temporairement dans la base de données lors de la migration de parties plus simples. Cette approche hybride préserve la stabilité du système pendant que vous progressez vers vos objectifs architecturaux. Surveillez en permanence les performances et les fonctionnalités pendant la migration, et soyez prêt à ajuster votre stratégie en fonction des résultats.

Comment gérer les déclencheurs de base de données lors de la migration ?

Transformez les déclencheurs de base de données en gestionnaires d'événements au niveau de l'application tout en préservant les fonctionnalités du système. Remplacez les déclencheurs

synchrones par des modèles pilotés par des événements qui envoient des messages aux files d'attente pour les opérations asynchrones. Envisagez d'utiliser Amazon Simple Notification Service (Amazon SNS) ou Amazon Simple Queue Service (Amazon SQS) pour les files d'attente de messages. Pour les exigences d'audit, implémentez la journalisation au niveau de l'application ou utilisez les fonctionnalités de capture des données de modification de base de données (CDC).

Analysez l'objectif et la criticité de chaque déclencheur. Certains déclencheurs peuvent être mieux servis par la logique de l'application, tandis que d'autres peuvent nécessiter des modèles d'approvisionnement en événements pour maintenir la cohérence des données. Commencez par des déclencheurs simples, tels que les journaux d'audit, avant de vous attaquer à des déclencheurs complexes qui gèrent les règles métier ou l'intégrité des données. Effectuez une surveillance attentive pendant la migration pour vous assurer qu'il n'y a aucune perte de fonctionnalité ou de cohérence des données.

Quel est le meilleur moyen de tester la logique métier migrée ?

Mettez en œuvre une approche de test à plusieurs niveaux avant de déployer la logique métier migrée. Commencez par des tests unitaires pour le nouveau code d'application, puis ajoutez des tests d'intégration qui couvrent les flux end-to-end métier. Exécutez les anciennes et les nouvelles implémentations en parallèle, puis comparez les résultats afin de valider l'équivalence fonctionnelle. Effectuez des tests de performance dans différentes conditions de charge pour vérifier que le comportement du système correspond ou dépasse les capacités précédentes.

Utilisez des indicateurs de fonctionnalité pour contrôler le déploiement afin de pouvoir revenir rapidement en arrière en cas de problème. Impliquez les utilisateurs professionnels dans la validation, en particulier pour les flux de travail critiques. Surveillez les indicateurs clés lors du déploiement initial et augmentez progressivement le trafic vers la nouvelle implémentation. Tout au long, conservez la possibilité de revenir à la logique de base de données d'origine si nécessaire.

Comment gérer la période de transition lorsque la logique de base de données et d'application existe à la fois ?

Lorsque la base de données et la logique de l'application sont toutes deux utilisées, implémentez des indicateurs de fonctionnalité qui contrôlent le flux de trafic et permettent de basculer rapidement entre les anciennes et les nouvelles implémentations. Maintenez un contrôle rigoureux des versions et documentez clairement les implémentations et leurs responsabilités respectives. Configurez une surveillance complète pour les deux systèmes afin d'identifier rapidement toute anomalie ou tout problème de performance.

Établissez des procédures d'annulation claires pour chaque composant migré afin de pouvoir revenir à la logique d'origine si nécessaire. Communiquez régulièrement avec toutes les parties prenantes sur l'état de la transition, les impacts potentiels et les procédures d'escalade. Cette approche vous permet de migrer progressivement tout en préservant la stabilité du système et la confiance des parties prenantes.

Comment gérer les scénarios d'erreur dans la couche d'application qui étaient auparavant gérés par la base de données ?

Remplacez la gestion des erreurs au niveau de la base de données par des mécanismes robustes au niveau de la couche application. Implémentez des disjoncteurs et réessayez une logique pour les défaillances transitoires. Utilisez des transactions de compensation pour maintenir la cohérence des données entre les opérations distribuées. Par exemple, si une mise à jour de paiement échoue, l'application doit automatiquement réessayer dans les limites définies et lancer des actions de compensation si nécessaire.

Configurez une surveillance et des alertes complètes pour identifier rapidement les problèmes et maintenez des journaux d'audit détaillés pour le dépannage. Concevez la gestion des erreurs de manière à ce qu'elle soit aussi automatisée que possible et définissez des voies d'escalade claires pour les scénarios nécessitant une intervention humaine. Cette approche multicouche assure la résilience du système tout en préservant l'intégrité des données et la continuité des processus métier.

Prochaines étapes pour la décomposition de la base de données sur AWS

Après avoir mis en œuvre des stratégies de décomposition de base de données initiales via des services d'encapsulation de base de données et transféré la logique métier vers la couche application, les entreprises doivent planifier leur prochaine évolution. Cette section décrit les principales considérations à prendre en compte pour poursuivre votre processus de modernisation.

Cette section contient les rubriques suivantes :

- Stratégies incrémentielles pour la décomposition des bases de données
- Considérations techniques relatives aux environnements de bases de données distribuées
- Changements organisationnels pour prendre en charge les architectures distribuées

Stratégies incrémentielles pour la décomposition des bases de données

La décomposition de la base de données suit une évolution progressive en trois phases distinctes. Les équipes intègrent d'abord la base de données monolithique à un service d'encapsulation de base de données pour contrôler l'accès. Ils commencent ensuite à diviser les données en bases de données spécifiques au service, tout en conservant la base de données principale pour répondre aux besoins existants. Enfin, ils effectuent la migration de la logique métier afin de passer à des bases de données de service totalement indépendantes.

Tout au long de ce parcours, les équipes doivent mettre en œuvre des modèles de synchronisation des données soignés et valider en permanence la cohérence entre les services. Le suivi des performances devient crucial pour identifier et résoudre les problèmes potentiels à un stade précoce. À mesure que les services évoluent indépendamment, leurs schémas doivent être optimisés en fonction des modèles d'utilisation réels, et vous devez supprimer les structures redondantes qui se sont accumulées au fil du temps.

Cette approche progressive permet de minimiser les risques tout en préservant la stabilité du système tout au long du processus de transformation.

Considérations techniques relatives aux environnements de bases de données distribuées

Dans un environnement de base de données distribuée, la surveillance des performances devient essentielle pour identifier et résoudre rapidement les goulots d'étranglement. Les équipes doivent mettre en œuvre des systèmes de surveillance complets et des stratégies de mise en cache pour maintenir les niveaux de performance. Read/write le fractionnement permet d'équilibrer efficacement les charges dans l'ensemble du système.

La cohérence des données nécessite une orchestration minutieuse entre les services distribués. Les équipes doivent mettre en œuvre d'éventuels modèles de cohérence, le cas échéant, et établir des limites claires en matière de propriété des données. Une surveillance robuste favorise l'intégrité des données dans tous les services.

En outre, la sécurité doit évoluer pour s'adapter à l'architecture distribuée. Chaque service nécessite des contrôles de sécurité précis, et vos modèles d'accès nécessitent un examen régulier. L'amélioration de la surveillance et de l'audit devient essentielle dans cet environnement distribué.

Changements organisationnels pour prendre en charge les architectures distribuées

La structure de l'équipe doit s'aligner sur les limites des services afin de définir clairement la propriété et la responsabilité. Organisations doivent établir de nouveaux modèles de communication et développer des capacités techniques supplémentaires au sein des équipes. Cette structure doit prendre en charge à la fois la maintenance des services existants et l'évolution continue de votre architecture.

Vous devez mettre à jour vos processus opérationnels pour gérer l'architecture distribuée. Les équipes doivent modifier les procédures de déploiement, adapter les processus de réponse aux incidents et faire évoluer les pratiques de gestion du changement afin de coordonner plusieurs services.

Ressources

Les ressources et outils supplémentaires suivants peuvent aider votre organisation dans le processus de décomposition de sa base de données.

AWS Conseils prescriptifs

- Migration de Oracle bases de données vers AWS Cloud
- Options de replateforme pour Oracle DatabaseAWS
- Modèles de conception, architectures et implémentations du cloud

AWS articles de blog

- Migrez la logique métier de la base de données vers l'application pour accélérer l'innovation et la flexibilité

Services AWS

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

Autres outils

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

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

Autres ressources

- Du monolithe aux microservices (site web) O'Reilly

Historique du document

Le tableau suivant décrit les modifications importantes apportées à ce guide. Pour être averti des mises à jour à venir, abonnez-vous à un fil RSS.

Modification	Description	Date
FAQ sur le mainframe et outils d'IA	Nous avons ajouté les recommandations s'appliquent-elles aux bases de données mainframe monolithiques ? FAQ, et nous avons ajouté des informations supplémentaires sur les outils d'IA que vous pouvez utiliser lors de la décomposition de la base de données.	14 octobre 2025
Publication initiale	—	30 septembre 2025

AWS Glossaire des directives prescriptives

Les termes suivants sont couramment utilisés dans les stratégies, les guides et les modèles fournis par les directives AWS prescriptives. Pour suggérer des entrées, veuillez utiliser le lien Faire un commentaire à la fin du glossaire.

Nombres

7 R

Sept politiques de migration courantes pour transférer des applications vers le cloud. Ces politiques s'appuient sur les 5 R identifiés par Gartner en 2011 et sont les suivantes :

- Refactorisation/réarchitecture : transférez une application et modifiez son architecture en tirant pleinement parti des fonctionnalités natives cloud pour améliorer l'agilité, les performances et la capacité de mise à l'échelle. Cela implique généralement le transfert du système d'exploitation et de la base de données. Exemple : migrez votre base de données Oracle sur site vers l'édition compatible avec Amazon Aurora PostgreSQL.
- Replateformer (déplacer et remodeler) : transférez une application vers le cloud et introduisez un certain niveau d'optimisation pour tirer parti des fonctionnalités du cloud. Exemple : migrez votre base de données Oracle sur site vers Amazon Relational Database Service (Amazon RDS) pour Oracle dans le. AWS Cloud
- Racheter (rachat) : optez pour un autre produit, généralement en passant d'une licence traditionnelle à un modèle SaaS. Exemple : migrez votre système de gestion de la relation client (CRM) vers Salesforce.com.
- Réhéberger (lift and shift) : transférez une application vers le cloud sans apporter de modifications pour tirer parti des fonctionnalités du cloud. Exemple : migrez votre base de données Oracle sur site vers Oracle sur une instance EC2 dans le. AWS Cloud
- Relocaliser (lift and shift au niveau de l'hyperviseur) : transférez l'infrastructure vers le cloud sans acheter de nouveau matériel, réécrire des applications ou modifier vos opérations existantes. Vous migrez des serveurs d'une plateforme sur site vers un service cloud pour la même plateforme. Exemple : migrer une Microsoft Hyper-V application vers AWS.
- Retenir : conservez les applications dans votre environnement source. Il peut s'agir d'applications nécessitant une refactorisation majeure, que vous souhaitez retarder, et d'applications existantes que vous souhaitez retenir, car rien ne justifie leur migration sur le plan commercial.

- Retirer : mettez hors service ou supprimez les applications dont vous n'avez plus besoin dans votre environnement source.

A

ABAC

Voir contrôle d'accès basé sur les attributs.

services abstraits

Consultez la section Services gérés.

ACIDE

Voir atomicité, consistance, isolation, durabilité.

migration active-active

Méthode de migration de base de données dans laquelle la synchronisation des bases de données source et cible est maintenue (à l'aide d'un outil de réplication bidirectionnelle ou d'opérations d'écriture double), tandis que les deux bases de données gèrent les transactions provenant de la connexion d'applications pendant la migration. Cette méthode prend en charge la migration par petits lots contrôlés au lieu d'exiger un basculement ponctuel. Elle est plus flexible mais demande plus de travail qu'une migration active-passive.

migration active-passive

Méthode de migration de base de données dans laquelle les bases de données source et cible sont synchronisées, mais seule la base de données source gère les transactions liées à la connexion des applications pendant que les données sont répliquées vers la base de données cible. La base de données cible n'accepte aucune transaction pendant la migration.

fonction d'agrégation

Fonction SQL qui agit sur un groupe de lignes et calcule une valeur de retour unique pour le groupe. Des exemples de fonctions d'agrégation incluent `SUM` et`MAX`.

AI

Voir intelligence artificielle.

AIOps

Voir les opérations d'intelligence artificielle.

anonymisation

Processus de suppression définitive d'informations personnelles dans un ensemble de données. L'anonymisation peut contribuer à protéger la vie privée. Les données anonymisées ne sont plus considérées comme des données personnelles.

anti-motif

Solution fréquemment utilisée pour un problème récurrent lorsque la solution est contre-productive, inefficace ou moins efficace qu'une alternative.

contrôle des applications

Une approche de sécurité qui permet d'utiliser uniquement des applications approuvées afin de protéger un système contre les logiciels malveillants.

portefeuille d'applications

Ensemble d'informations détaillées sur chaque application utilisée par une organisation, y compris le coût de génération et de maintenance de l'application, ainsi que sa valeur métier. Ces informations sont essentielles pour le processus de découverte et d'analyse du portefeuille et permettent d'identifier et de prioriser les applications à migrer, à moderniser et à optimiser.

intelligence artificielle (IA)

Domaine de l'informatique consacré à l'utilisation des technologies de calcul pour exécuter des fonctions cognitives généralement associées aux humains, telles que l'apprentissage, la résolution de problèmes et la reconnaissance de modèles. Pour plus d'informations, veuillez consulter Qu'est-ce que l'intelligence artificielle ?

opérations d'intelligence artificielle (AIOps)

Processus consistant à utiliser des techniques de machine learning pour résoudre les problèmes opérationnels, réduire les incidents opérationnels et les interventions humaines, mais aussi améliorer la qualité du service. Pour plus d'informations sur son AIOps utilisation dans la stratégie de AWS migration, consultez le guide d'intégration des opérations.

chiffrement asymétrique

Algorithme de chiffrement qui utilise une paire de clés, une clé publique pour le chiffrement et une clé privée pour le déchiffrement. Vous pouvez partager la clé publique, car elle n'est pas utilisée pour le déchiffrement, mais l'accès à la clé privée doit être très restreint.

atomicité, cohérence, isolement, durabilité (ACID)

Ensemble de propriétés logicielles garantissant la validité des données et la fiabilité opérationnelle d'une base de données, même en cas d'erreur, de panne de courant ou d'autres problèmes.

contrôle d'accès par attributs (ABAC)

Pratique qui consiste à créer des autorisations détaillées en fonction des attributs de l'utilisateur, tels que le service, le poste et le nom de l'équipe. Pour plus d'informations, consultez ABAC pour AWS dans la documentation AWS Identity and Access Management (IAM).

source de données faisant autorité

Emplacement où vous stockez la version principale des données, considérée comme la source d'information la plus fiable. Vous pouvez copier les données de la source de données officielle vers d'autres emplacements à des fins de traitement ou de modification des données, par exemple en les anonymisant, en les expurgant ou en les pseudonymisant.

Zone de disponibilité

Un emplacement distinct au sein d'un Région AWS réseau isolé des défaillances dans d'autres zones de disponibilité et fournissant une connectivité réseau peu coûteuse et à faible latence aux autres zones de disponibilité de la même région.

AWS Cadre d'adoption du cloud (AWS CAF)

Un cadre de directives et de meilleures pratiques visant AWS à aider les entreprises à élaborer un plan efficace pour réussir leur migration vers le cloud. AWS La CAF organise ses conseils en six domaines prioritaires appelés perspectives : les affaires, les personnes, la gouvernance, les plateformes, la sécurité et les opérations. Les perspectives d'entreprise, de personnes et de gouvernance mettent l'accent sur les compétences et les processus métier, tandis que les perspectives relatives à la plateforme, à la sécurité et aux opérations se concentrent sur les compétences et les processus techniques. Par exemple, la perspective liée aux personnes cible les parties prenantes qui s'occupent des ressources humaines (RH), des fonctions de dotation en personnel et de la gestion des personnes. Dans cette perspective, la AWS CAF fournit des conseils pour le développement du personnel, la formation et les communications afin de préparer l'organisation à une adoption réussie du cloud. Pour plus d'informations, veuillez consulter le site Web AWS CAF et le livre blanc AWS CAF.

AWS Cadre de qualification de la charge de travail (AWS WQF)

Outil qui évalue les charges de travail liées à la migration des bases de données, recommande des stratégies de migration et fournit des estimations de travail. AWS Le WQF est inclus avec

AWS Schema Conversion Tool (AWS SCT). Il analyse les schémas de base de données et les objets de code, le code d'application, les dépendances et les caractéristiques de performance, et fournit des rapports d'évaluation.

B

mauvais bot

Un bot destiné à perturber ou à nuire à des individus ou à des organisations.

BCP

Consultez la section Planification de la continuité des activités.

graphique de comportement

Vue unifiée et interactive des comportements des ressources et des interactions au fil du temps. Vous pouvez utiliser un graphique de comportement avec Amazon Detective pour examiner les tentatives de connexion infructueuses, les appels d'API suspects et les actions similaires. Pour plus d'informations, veuillez consulter Data in a behavior graph dans la documentation Detective.

système de poids fort

Système qui stocke d'abord l'octet le plus significatif. Voir aussi endianité.

classification binaire

Processus qui prédit un résultat binaire (l'une des deux classes possibles). Par exemple, votre modèle de machine learning peut avoir besoin de prévoir des problèmes tels que « Cet e-mail est-il du spam ou non ? » ou « Ce produit est-il un livre ou une voiture ? ».

filtre de Bloom

Structure de données probabiliste et efficace en termes de mémoire qui est utilisée pour tester si un élément fait partie d'un ensemble.

déploiement bleu/vert

Stratégie de déploiement dans laquelle vous créez deux environnements distincts mais identiques. Vous exécutez la version actuelle de l'application dans un environnement (bleu) et la nouvelle version de l'application dans l'autre environnement (vert). Cette stratégie vous permet de revenir rapidement en arrière avec un impact minimal.

bot

Application logicielle qui exécute des tâches automatisées sur Internet et simule l'activité ou l'interaction humaine. Certains robots sont utiles ou bénéfiques, comme les robots d'exploration Web qui indexent des informations sur Internet. D'autres robots, appelés « bots malveillants », sont destinés à perturber ou à nuire à des individus ou à des organisations.

botnet

Réseaux de robots infectés par des logiciels malveillants et contrôlés par une seule entité, connue sous le nom d'herder ou d'opérateur de bots. Les botnets sont le mécanisme le plus connu pour faire évoluer les bots et leur impact.

branche

Zone contenue d'un référentiel de code. La première branche créée dans un référentiel est la branche principale. Vous pouvez créer une branche à partir d'une branche existante, puis développer des fonctionnalités ou corriger des bogues dans la nouvelle branche. Une branche que vous créez pour générer une fonctionnalité est communément appelée branche de fonctionnalités. Lorsque la fonctionnalité est prête à être publiée, vous fusionnez à nouveau la branche de fonctionnalités dans la branche principale. Pour plus d'informations, consultez À propos des branches (GitHub documentation).

accès par brise-vitre

Dans des circonstances exceptionnelles et par le biais d'un processus approuvé, c'est un moyen rapide pour un utilisateur d'accéder à un accès auquel Compte AWS il n'est généralement pas autorisé. Pour plus d'informations, consultez l'indicateur Implementation break-glass procedures dans le guide Well-Architected AWS .

stratégie existante (brownfield)

L'infrastructure existante de votre environnement. Lorsque vous adoptez une stratégie existante pour une architecture système, vous concevez l'architecture en fonction des contraintes des systèmes et de l'infrastructure actuels. Si vous étendez l'infrastructure existante, vous pouvez combiner des politiques brownfield (existantes) et greenfield (inédites).

cache de tampon

Zone de mémoire dans laquelle sont stockées les données les plus fréquemment consultées.

capacité métier

Ce que fait une entreprise pour générer de la valeur (par exemple, les ventes, le service client ou le marketing). Les architectures de microservices et les décisions de développement

peuvent être dictées par les capacités métier. Pour plus d'informations, veuillez consulter la section Organisation en fonction des capacités métier du livre blanc Exécution de microservices conteneurisés sur AWS.

planification de la continuité des activités (BCP)

Plan qui tient compte de l'impact potentiel d'un événement perturbateur, tel qu'une migration à grande échelle, sur les opérations, et qui permet à une entreprise de reprendre ses activités rapidement.

C

CAF

Voir le cadre d'adoption du AWS cloud.

déploiement de Canary

Diffusion lente et progressive d'une version pour les utilisateurs finaux. Lorsque vous êtes sûr, vous déployez la nouvelle version et remplacez la version actuelle dans son intégralité.

CCo E

Voir le Centre d'excellence du cloud.

CDC

Voir capture des données de modification.

capture des données de modification (CDC)

Processus de suivi des modifications apportées à une source de données, telle qu'une table de base de données, et d'enregistrement des métadonnées relatives à ces modifications. Vous pouvez utiliser la CDC à diverses fins, telles que l'audit ou la réplication des modifications dans un système cible afin de maintenir la synchronisation.

ingénierie du chaos

Introduire intentionnellement des défaillances ou des événements perturbateurs pour tester la résilience d'un système. Vous pouvez utiliser AWS Fault Injection Service (AWS FIS) pour effectuer des expériences qui stressent vos AWS charges de travail et évaluer leur réponse.

CI/CD

Découvrez l'intégration continue et la livraison continue.

classification

Processus de catégorisation qui permet de générer des prédictions. Les modèles de ML pour les problèmes de classification prédisent une valeur discrète. Les valeurs discrètes se distinguent toujours les unes des autres. Par exemple, un modèle peut avoir besoin d'évaluer la présence ou non d'une voiture sur une image.

chiffrement côté client

Chiffrement des données localement, avant que la cible ne les Service AWS reçoive.

Centre d'excellence du cloud (CCoE)

Une équipe multidisciplinaire qui dirige les efforts d'adoption du cloud au sein d'une organisation, notamment en développant les bonnes pratiques en matière de cloud, en mobilisant des ressources, en établissant des délais de migration et en guidant l'organisation dans le cadre de transformations à grande échelle. Pour plus d'informations, consultez les CCoarticles électroniques du blog sur la stratégie AWS Cloud d'entreprise.

cloud computing

Technologie cloud généralement utilisée pour le stockage de données à distance et la gestion des appareils IoT. Le cloud computing est généralement associé à la technologie informatique de pointe.

modèle d'exploitation du cloud

Dans une organisation informatique, modèle d'exploitation utilisé pour créer, faire évoluer et optimiser un ou plusieurs environnements cloud. Pour plus d'informations, consultez la section Création de votre modèle d'exploitation cloud.

étapes d'adoption du cloud

Les quatre phases que les entreprises traversent généralement lorsqu'elles migrent vers AWS Cloud :

- Projet : exécution de quelques projets liés au cloud à des fins de preuve de concept et d'apprentissage
- Base : réaliser des investissements fondamentaux pour accélérer votre adoption du cloud (par exemple, créer une zone de landing zone, définir un CCo E, établir un modèle opérationnel)
- Migration : migration d'applications individuelles
- Réinvention : optimisation des produits et services et innovation dans le cloud

Ces étapes ont été définies par Stephen Orban dans le billet de blog The Journey Toward Cloud-First & the Stages of Adoption publié sur le blog AWS Cloud Enterprise Strategy. Pour plus d'informations sur leur lien avec la stratégie de AWS migration, consultez le guide de préparation à la migration.

CMDB

Consultez la base de données de gestion des configurations.

référentiel de code

Emplacement où le code source et d'autres ressources, comme la documentation, les exemples et les scripts, sont stockés et mis à jour par le biais de processus de contrôle de version. Les référentiels cloud courants incluent GitHub ouBitbucket Cloud. Chaque version du code est appelée branche. Dans une structure de microservice, chaque référentiel est consacré à une seule fonctionnalité. Un seul pipeline CI/CD peut utiliser plusieurs référentiels.

cache passif

Cache tampon vide, mal rempli ou contenant des données obsolètes ou non pertinentes. Cela affecte les performances, car l'instance de base de données doit lire à partir de la mémoire principale ou du disque, ce qui est plus lent que la lecture à partir du cache tampon.

données gelées

Données rarement consultées et généralement historiques. Lorsque vous interrogez ce type de données, les requêtes lentes sont généralement acceptables. Le transfert de ces données vers des niveaux ou classes de stockage moins performants et moins coûteux peut réduire les coûts.

vision par ordinateur (CV)

Domaine de l'IA qui utilise l'apprentissage automatique pour analyser et extraire des informations à partir de formats visuels tels que des images numériques et des vidéos. Par exemple, Amazon SageMaker AI fournit des algorithmes de traitement d'image pour les CV.

dérive de configuration

Pour une charge de travail, une modification de configuration par rapport à l'état attendu. Cela peut entraîner une non-conformité de la charge de travail, et cela est généralement progressif et involontaire.

base de données de gestion des configurations (CMDB)

Référentiel qui stocke et gère les informations relatives à une base de données et à son environnement informatique, y compris les composants matériels et logiciels ainsi que leurs

configurations. Vous utilisez généralement les données d'une CMDB lors de la phase de découverte et d'analyse du portefeuille de la migration.

pack de conformité

Ensemble de AWS Config règles et d'actions correctives que vous pouvez assembler pour personnaliser vos contrôles de conformité et de sécurité. Vous pouvez déployer un pack de conformité en tant qu'entité unique dans une région Compte AWS et, ou au sein d'une organisation, à l'aide d'un modèle YAML. Pour plus d'informations, consultez la section Packs de conformité dans la AWS Config documentation.

intégration continue et livraison continue (CI/CD)

Processus d'automatisation des étapes de source, de construction, de test, de préparation et de production du processus de publication du logiciel. CI/CD est communément décrit comme un pipeline. CI/CD peut vous aider à automatiser les processus, à améliorer la productivité, à améliorer la qualité du code et à accélérer les livraisons. Pour plus d'informations, veuillez consulter Avantages de la livraison continue. CD peut également signifier déploiement continu. Pour plus d'informations, veuillez consulter Livraison continue et déploiement continu.

CV

Voir vision par ordinateur.

D

données au repos

Données stationnaires dans votre réseau, telles que les données stockées.

classification des données

Processus permettant d'identifier et de catégoriser les données de votre réseau en fonction de leur sévérité et de leur sensibilité. Il s'agit d'un élément essentiel de toute stratégie de gestion des risques de cybersécurité, car il vous aide à déterminer les contrôles de protection et de conservation appropriés pour les données. La classification des données est une composante du pilier de sécurité du AWS Well-Architected Framework. Pour plus d'informations, veuillez consulter Classification des données.

dérive des données

Une variation significative entre les données de production et les données utilisées pour entraîner un modèle ML, ou une modification significative des données d'entrée au fil du temps. La dérive

des données peut réduire la qualité, la précision et l'équité globales des prédictions des modèles ML.

données en transit

Données qui circulent activement sur votre réseau, par exemple entre les ressources du réseau.

maillage de données

Un cadre architectural qui fournit une propriété des données distribuée et décentralisée avec une gestion et une gouvernance centralisées.

minimisation des données

Le principe de collecte et de traitement des seules données strictement nécessaires. La pratique de la minimisation des données AWS Cloud peut réduire les risques liés à la confidentialité, les coûts et l'empreinte carbone de vos analyses.

périmètre de données

Ensemble de garde-fous préventifs dans votre AWS environnement qui permettent de garantir que seules les identités fiables accèdent aux ressources fiables des réseaux attendus. Pour plus d'informations, voir Création d'un périmètre de données sur AWS.

prétraitement des données

Pour transformer les données brutes en un format facile à analyser par votre modèle de ML. Le prétraitement des données peut impliquer la suppression de certaines colonnes ou lignes et le traitement des valeurs manquantes, incohérentes ou en double.

provenance des données

Le processus de suivi de l'origine et de l'historique des données tout au long de leur cycle de vie, par exemple la manière dont les données ont été générées, transmises et stockées.

sujet des données

Personne dont les données sont collectées et traitées.

entrepôt des données

Un système de gestion des données qui prend en charge les informations commerciales, telles que les analyses. Les entrepôts de données contiennent généralement de grandes quantités de données historiques et sont généralement utilisés pour les requêtes et les analyses.

langage de définition de base de données (DDL)

Instructions ou commandes permettant de créer ou de modifier la structure des tables et des objets dans une base de données.

langage de manipulation de base de données (DML)

Instructions ou commandes permettant de modifier (insérer, mettre à jour et supprimer) des informations dans une base de données.

DDL

Voir langage de définition de base de données.

ensemble profond

Sert à combiner plusieurs modèles de deep learning à des fins de prédiction. Vous pouvez utiliser des ensembles profonds pour obtenir une prévision plus précise ou pour estimer l'incertitude des prédictions.

deep learning

Un sous-champ de ML qui utilise plusieurs couches de réseaux neuronaux artificiels pour identifier le mappage entre les données d'entrée et les variables cibles d'intérêt.

defense-in-depth

Approche de la sécurité de l'information dans laquelle une série de mécanismes et de contrôles de sécurité sont judicieusement répartis sur l'ensemble d'un réseau informatique afin de protéger la confidentialité, l'intégrité et la disponibilité du réseau et des données qu'il contient. Lorsque vous adoptez cette stratégie AWS, vous ajoutez plusieurs contrôles à différentes couches de la AWS Organizations structure afin de sécuriser les ressources. Par exemple, une defense-in-depth approche peut combiner l'authentification multifactorielle, la segmentation du réseau et le chiffrement.

administrateur délégué

Dans AWS Organizations, un service compatible peut enregistrer un compte AWS membre pour administrer les comptes de l'organisation et gérer les autorisations pour ce service. Ce compte est appelé administrateur délégué pour ce service. Pour plus d'informations et une liste des services compatibles, veuillez consulter la rubrique Services qui fonctionnent avec AWS Organizations dans la documentation AWS Organizations .

déploiement

Processus de mise à disposition d'une application, de nouvelles fonctionnalités ou de corrections de code dans l'environnement cible. Le déploiement implique la mise en œuvre de modifications dans une base de code, puis la génération et l'exécution de cette base de code dans les environnements de l'application.

environnement de développement

Voir environnement.

contrôle de détection

Contrôle de sécurité conçu pour détecter, journaliser et alerter après la survenue d'un événement. Ces contrôles constituent une deuxième ligne de défense et vous alertent en cas d'événements de sécurité qui ont contourné les contrôles préventifs en place. Pour plus d'informations, veuillez consulter la rubrique Contrôles de détection dans Implementing security controls on AWS.

cartographie de la chaîne de valeur du développement (DVSM)

Processus utilisé pour identifier et hiérarchiser les contraintes qui nuisent à la rapidité et à la qualité du cycle de vie du développement logiciel. DVSM étend le processus de cartographie de la chaîne de valeur initialement conçu pour les pratiques de production allégée. Il met l'accent sur les étapes et les équipes nécessaires pour créer et transférer de la valeur tout au long du processus de développement logiciel.

jumeau numérique

Représentation virtuelle d'un système réel, tel qu'un bâtiment, une usine, un équipement industriel ou une ligne de production. Les jumeaux numériques prennent en charge la maintenance prédictive, la surveillance à distance et l'optimisation de la production.

tableau des dimensions

Dans un schéma en étoile, table plus petite contenant les attributs de données relatifs aux données quantitatives d'une table de faits. Les attributs des tables de dimensions sont généralement des champs de texte ou des nombres discrets qui se comportent comme du texte. Ces attributs sont couramment utilisés pour la contrainte des requêtes, le filtrage et l'étiquetage des ensembles de résultats.

catastrophe

Un événement qui empêche une charge de travail ou un système d'atteindre ses objectifs commerciaux sur son site de déploiement principal. Ces événements peuvent être des

catastrophes naturelles, des défaillances techniques ou le résultat d'actions humaines, telles qu'une mauvaise configuration involontaire ou une attaque de logiciel malveillant.

reprise après sinistre (DR)

La stratégie et le processus que vous utilisez pour minimiser les temps d'arrêt et les pertes de données causés par un sinistre. Pour plus d'informations, consultez Disaster Recovery of Workloads on AWS : Recovery in the Cloud in the AWS Well-Architected Framework.

DML

Voir langage de manipulation de base de données.

conception axée sur le domaine

Approche visant à développer un système logiciel complexe en connectant ses composants à des domaines évolutifs, ou objectifs métier essentiels, que sert chaque composant. Ce concept a été introduit par Eric Evans dans son ouvrage Domain-Driven Design: Tackling Complexity in the Heart of Software (Boston : Addison-Wesley Professional, 2003). Pour plus d'informations sur l'utilisation du design piloté par domaine avec le modèle de figuier étrangleur, veuillez consulter Modernizing legacy Microsoft ASP.NET (ASMX) web services incrementally by using containers and Amazon API Gateway.

DR

Voir reprise après sinistre.

détection de dérive

Suivi des écarts par rapport à une configuration de référence. Par exemple, vous pouvez l'utiliser AWS CloudFormation pour détecter la dérive des ressources du système ou AWS Control Tower pour détecter les modifications de votre zone d'atterrissage susceptibles d'affecter le respect des exigences de gouvernance.

DVSM

Voir la cartographie de la chaîne de valeur du développement.

E

EDA

Voir analyse exploratoire des données.

EDI

Voir échange de données informatisé.

informatique de périphérie

Technologie qui augmente la puissance de calcul des appareils intelligents en périphérie d'un réseau IoT. Comparé au cloud computing, l'informatique de pointe peut réduire la latence des communications et améliorer le temps de réponse.

échange de données informatisé (EDI)

L'échange automatique de documents commerciaux entre les organisations. Pour plus d'informations, voir Qu'est-ce que l'échange de données informatisé ?

chiffrement

Processus informatique qui transforme des données en texte clair, lisibles par l'homme, en texte chiffré.

clé de chiffrement

Chaîne cryptographique de bits aléatoires générée par un algorithme cryptographique. La longueur des clés peut varier, et chaque clé est conçue pour être imprévisible et unique.

endianisme

Ordre selon lequel les octets sont stockés dans la mémoire de l'ordinateur. Les systèmes de poids fort stockent d'abord l'octet le plus significatif. Les systèmes de poids faible stockent d'abord l'octet le moins significatif.

point de terminaison

Voir point de terminaison de service.

service de point de terminaison

Service que vous pouvez héberger sur un cloud privé virtuel (VPC) pour le partager avec d'autres utilisateurs. Vous pouvez créer un service de point de terminaison avec AWS PrivateLink et accorder des autorisations à d'autres Comptes AWS ou à AWS Identity and Access Management (IAM) principaux. Ces comptes ou principaux peuvent se connecter à votre service de point de terminaison de manière privée en créant des points de terminaison d'un VPC d'interface. Pour plus d'informations, veuillez consulter Création d'un service de point de terminaison dans la documentation Amazon Virtual Private Cloud (Amazon VPC).

planification des ressources d'entreprise (ERP)

Système qui automatise et gère les principaux processus métier (tels que la comptabilité, le MES et la gestion de projet) pour une entreprise.

chiffrement d'enveloppe

Processus de chiffrement d'une clé de chiffrement à l'aide d'une autre clé de chiffrement. Pour plus d'informations, consultez la section Chiffrement des enveloppes dans la documentation AWS Key Management Service (AWS KMS).

environment

Instance d'une application en cours d'exécution. Les types d'environnement les plus courants dans le cloud computing sont les suivants :

- Environnement de développement : instance d'une application en cours d'exécution à laquelle seule l'équipe principale chargée de la maintenance de l'application peut accéder. Les environnements de développement sont utilisés pour tester les modifications avant de les promouvoir dans les environnements supérieurs. Ce type d'environnement est parfois appelé environnement de test.
- Environnements inférieurs : tous les environnements de développement d'une application, tels que ceux utilisés pour les générations et les tests initiaux.
- Environnement de production : instance d'une application en cours d'exécution à laquelle les utilisateurs finaux peuvent accéder. Dans un CI/CD pipeline, l'environnement de production est le dernier environnement de déploiement.
- Environnements supérieurs : tous les environnements accessibles aux utilisateurs autres que l'équipe de développement principale. Ils peuvent inclure un environnement de production, des environnements de préproduction et des environnements pour les tests d'acceptation par les utilisateurs.

épopée

Dans les méthodologies agiles, catégories fonctionnelles qui aident à organiser et à prioriser votre travail. Les épopées fournissent une description détaillée des exigences et des tâches d'implémentation. Par exemple, les points forts de la AWS CAF en matière de sécurité incluent la gestion des identités et des accès, les contrôles de détection, la sécurité des infrastructures, la protection des données et la réponse aux incidents. Pour plus d'informations sur les épopées dans la stratégie de migration AWS , veuillez consulter le guide d'implémentation du programme.

ERP

Voir Planification des ressources d'entreprise.

analyse exploratoire des données (EDA)

Processus d'analyse d'un jeu de données pour comprendre ses principales caractéristiques. Vous collectez ou agrégez des données, puis vous effectuez des enquêtes initiales pour trouver des modèles, détecter des anomalies et vérifier les hypothèses. L'EDA est réalisée en calculant des statistiques récapitulatives et en créant des visualisations de données.

F

tableau des faits

La table centrale dans un schéma en étoile. Il stocke des données quantitatives sur les opérations commerciales. Généralement, une table de faits contient deux types de colonnes : celles qui contiennent des mesures et celles qui contiennent une clé étrangère pour une table de dimensions.

échouer rapidement

Une philosophie qui utilise des tests fréquents et progressifs pour réduire le cycle de vie du développement. C'est un élément essentiel d'une approche agile.

limite d'isolation des défauts

Dans le AWS Cloud, une limite telle qu'une zone de disponibilité Région AWS, un plan de contrôle ou un plan de données qui limite l'effet d'une panne et contribue à améliorer la résilience des charges de travail. Pour plus d'informations, consultez la section Limites d'isolation des AWS pannes.

branche de fonctionnalités

Voir succursale.

fonctionnalités

Les données d'entrée que vous utilisez pour faire une prédiction. Par exemple, dans un contexte de fabrication, les fonctionnalités peuvent être des images capturées périodiquement à partir de la ligne de fabrication.

importance des fonctionnalités

Le niveau d'importance d'une fonctionnalité pour les prédictions d'un modèle. Il s'exprime généralement sous la forme d'un score numérique qui peut être calculé à l'aide de différentes techniques, telles que la méthode Shapley Additive Explanations (SHAP) et les gradients intégrés. Pour plus d'informations, voir Interprétabilité du modèle d'apprentissage automatique avec AWS.

transformation de fonctionnalité

Optimiser les données pour le processus de ML, notamment en enrichissant les données avec des sources supplémentaires, en mettant à l'échelle les valeurs ou en extrayant plusieurs ensembles d'informations à partir d'un seul champ de données. Cela permet au modèle de ML de tirer parti des données. Par exemple, si vous décomposez la date « 2021-05-27 00:15:37 » en « 2021 », « mai », « jeudi » et « 15 », vous pouvez aider l'algorithme d'apprentissage à apprendre des modèles nuancés associés à différents composants de données.

invitation en quelques coups

Fournir à un LLM un petit nombre d'exemples illustrant la tâche et le résultat souhaité avant de lui demander d'effectuer une tâche similaire. Cette technique est une application de l'apprentissage contextuel, dans le cadre de laquelle les modèles apprennent à partir d'exemples (prises de vue) intégrés dans des instructions. Les instructions en quelques étapes peuvent être efficaces pour les tâches qui nécessitent un formatage, un raisonnement ou des connaissances de domaine spécifiques. Voir également l'invite Zero-Shot.

FGAC

Découvrez le contrôle d'accès détaillé.

contrôle d'accès détaillé (FGAC)

Utilisation de plusieurs conditions pour autoriser ou refuser une demande d'accès.

migration instantanée (flash-cut)

Méthode de migration de base de données qui utilise la réplication continue des données par le biais de la capture des données de modification afin de migrer les données dans les plus brefs délais, au lieu d'utiliser une approche progressive. L'objectif est de réduire au maximum les temps d'arrêt.

FM

Voir le modèle de fondation.

modèle de fondation (FM)

Un vaste réseau neuronal d'apprentissage profond qui s'est entraîné sur d'énormes ensembles de données généralisées et non étiquetées. FMs sont capables d'effectuer une grande variété de tâches générales, telles que comprendre le langage, générer du texte et des images et converser en langage naturel. Pour plus d'informations, voir Que sont les modèles de base ?

G

IA générative

Sous-ensemble de modèles d'IA qui ont été entraînés sur de grandes quantités de données et qui peuvent utiliser une simple invite textuelle pour créer de nouveaux contenus et artefacts, tels que des images, des vidéos, du texte et du son. Pour plus d'informations, consultez Qu'est-ce que l'IA générative.

blocage géographique

Voir les restrictions géographiques.

restrictions géographiques (blocage géographique)

Sur Amazon CloudFront, option permettant d'empêcher les utilisateurs de certains pays d'accéder aux distributions de contenu. Vous pouvez utiliser une liste d'autorisation ou une liste de blocage pour spécifier les pays approuvés et interdits. Pour plus d'informations, consultez la section Restreindre la distribution géographique de votre contenu dans la CloudFront documentation.

Flux de travail Gitflow

Approche dans laquelle les environnements inférieurs et supérieurs utilisent différentes branches dans un référentiel de code source. Le flux de travail Gitflow est considéré comme existant, et le flux de travail basé sur les troncs est l'approche moderne préférée.

image dorée

Un instantané d'un système ou d'un logiciel utilisé comme modèle pour déployer de nouvelles instances de ce système ou logiciel. Par exemple, dans le secteur de la fabrication, une image dorée peut être utilisée pour fournir des logiciels sur plusieurs appareils et contribue à améliorer la vitesse, l'évolutivité et la productivité des opérations de fabrication des appareils.

stratégie inédite

L'absence d'infrastructures existantes dans un nouvel environnement. Lorsque vous adoptez une stratégie inédite pour une architecture système, vous pouvez sélectionner toutes les nouvelles technologies sans restriction de compatibilité avec l'infrastructure existante, également appelée brownfield. Si vous étendez l'infrastructure existante, vous pouvez combiner des politiques brownfield (existantes) et greenfield (inédites).

barrière de protection

Règle de haut niveau qui permet de régir les ressources, les politiques et la conformité au sein des unités organisationnelles (OUs). Les barrières de protection préventives appliquent des politiques pour garantir l'alignement sur les normes de conformité. Elles sont mises en œuvre à l'aide de politiques de contrôle des services et de limites des autorisations IAM. Les barrières de protection de détection détectent les violations des politiques et les problèmes de conformité, et génèrent des alertes pour y remédier. Ils sont implémentés à l'aide d'Amazon AWS Config AWS Security Hub CSPM GuardDuty AWS Trusted Advisor, d'Amazon Inspector et de AWS Lambda contrôles personnalisés.

H

HA

Découvrez la haute disponibilité.

migration de base de données hétérogène

Migration de votre base de données source vers une base de données cible qui utilise un moteur de base de données différent (par exemple, Oracle vers Amazon Aurora). La migration hétérogène fait généralement partie d'un effort de réarchitecture, et la conversion du schéma peut s'avérer une tâche complexe. AWS propose AWS SCT qui facilite les conversions de schémas.

haute disponibilité (HA)

Capacité d'une charge de travail à fonctionner en continu, sans intervention, en cas de difficultés ou de catastrophes. Les systèmes HA sont conçus pour basculer automatiquement, fournir constamment des performances de haute qualité et gérer différentes charges et défaillances avec un impact minimal sur les performances.

modernisation des historiens

Approche utilisée pour moderniser et mettre à niveau les systèmes de technologie opérationnelle (OT) afin de mieux répondre aux besoins de l'industrie manufacturière. Un historien est un type de base de données utilisé pour collecter et stocker des données provenant de diverses sources dans une usine.

données de rétention

Partie de données historiques étiquetées qui n'est pas divulguée dans un ensemble de données utilisé pour entraîner un modèle d'apprentissage automatique. Vous pouvez utiliser les données de blocage pour évaluer les performances du modèle en comparant les prévisions du modèle aux données de blocage.

migration de base de données homogène

Migration de votre base de données source vers une base de données cible qui partage le même moteur de base de données (par exemple, Microsoft SQL Server vers Amazon RDS for SQL Server). La migration homogène s'inscrit généralement dans le cadre d'un effort de réhébergement ou de replateforme. Vous pouvez utiliser les utilitaires de base de données natifs pour migrer le schéma.

données chaudes

Données fréquemment consultées, telles que les données en temps réel ou les données translationnelles récentes. Ces données nécessitent généralement un niveau ou une classe de stockage à hautes performances pour fournir des réponses rapides aux requêtes.

correctif

Solution d'urgence à un problème critique dans un environnement de production. En raison de son urgence, un correctif est généralement créé en dehors du flux de travail de DevOps publication habituel.

période de soins intensifs

Immédiatement après le basculement, période pendant laquelle une équipe de migration gère et surveille les applications migrées dans le cloud afin de résoudre les problèmes éventuels. En règle générale, cette période dure de 1 à 4 jours. À la fin de la période de soins intensifs, l'équipe de migration transfère généralement la responsabilité des applications à l'équipe des opérations cloud.

I

IaC

Considérez l'infrastructure comme un code.

politique basée sur l’identité

Politique attachée à un ou plusieurs principaux IAM qui définit leurs autorisations au sein de l' AWS Cloud environnement.

application inactive

Application dont l'utilisation moyenne du processeur et de la mémoire se situe entre 5 et 20 % sur une période de 90 jours. Dans un projet de migration, il est courant de retirer ces applications ou de les retenir sur site.

IIo T

Voir Internet industriel des objets.

infrastructure immuable

Modèle qui déploie une nouvelle infrastructure pour les charges de travail de production au lieu de mettre à jour, d'appliquer des correctifs ou de modifier l'infrastructure existante. Les infrastructures immuables sont intrinsèquement plus cohérentes, fiables et prévisibles que les infrastructures mutables. Pour plus d'informations, consultez les meilleures pratiques de déploiement à l'aide d'une infrastructure immuable dans le AWS Well-Architected Framework.

VPC entrant (d'entrée)

Dans une architecture AWS multi-comptes, un VPC qui accepte, inspecte et achemine les connexions réseau depuis l'extérieur d'une application. L'architecture AWS de référence de sécurité recommande de configurer votre compte réseau avec les fonctions entrantes, sortantes et d'inspection VPCs afin de protéger l'interface bidirectionnelle entre votre application et l'Internet en général.

migration incrémentielle

Stratégie de basculement dans le cadre de laquelle vous migrez votre application par petites parties au lieu d'effectuer un basculement complet unique. Par exemple, il se peut que vous ne transfériez que quelques microservices ou utilisateurs vers le nouveau système dans un premier temps. Après avoir vérifié que tout fonctionne correctement, vous pouvez transférer

progressivement des microservices ou des utilisateurs supplémentaires jusqu'à ce que vous puissiez mettre hors service votre système hérité. Cette stratégie réduit les risques associés aux migrations de grande ampleur.

Industry 4.0

Terme introduit par Klaus Schwab en 2016 pour désigner la modernisation des processus de fabrication grâce aux avancées en matière de connectivité, de données en temps réel, d'automatisation, d'analyse et d'IA/ML.

infrastructure

Ensemble des ressources et des actifs contenus dans l'environnement d'une application.

infrastructure en tant que code (IaC)

Processus de mise en service et de gestion de l'infrastructure d'une application via un ensemble de fichiers de configuration. IaC est conçue pour vous aider à centraliser la gestion de l'infrastructure, à normaliser les ressources et à mettre à l'échelle rapidement afin que les nouveaux environnements soient reproductibles, fiables et cohérents.

Internet industriel des objets (IIoT)

L'utilisation de capteurs et d'appareils connectés à Internet dans les secteurs industriels tels que la fabrication, l'énergie, l'automobile, les soins de santé, les sciences de la vie et l'agriculture. Pour plus d'informations, voir Élaboration d'une stratégie de transformation numérique de l'Internet des objets (IIoT) industriel.

VPC d'inspection

Dans une architecture AWS multi-comptes, un VPC centralisé qui gère les inspections du trafic réseau VPCs entre (identique ou Régions AWS différent), Internet et les réseaux locaux. L'architecture AWS de référence de sécurité recommande de configurer votre compte réseau avec les fonctions entrantes, sortantes et d'inspection VPCs afin de protéger l'interface bidirectionnelle entre votre application et l'Internet en général.

Internet des objets (IoT)

Réseau d'objets physiques connectés dotés de capteurs ou de processeurs intégrés qui communiquent avec d'autres appareils et systèmes via Internet ou via un réseau de communication local. Pour plus d'informations, veuillez consulter la section Qu'est-ce que l'IoT ?.

interprétabilité

Caractéristique d'un modèle de machine learning qui décrit dans quelle mesure un être humain peut comprendre comment les prédictions du modèle dépendent de ses entrées. Pour plus d'informations, voir Interprétabilité du modèle d'apprentissage automatique avec AWS.

IoT

Voir Internet des objets.

Bibliothèque d'informations informatiques (ITIL)

Ensemble de bonnes pratiques pour proposer des services informatiques et les aligner sur les exigences métier. L'ITIL constitue la base de l'ITSM.

gestion des services informatiques (ITSM)

Activités associées à la conception, à la mise en œuvre, à la gestion et à la prise en charge de services informatiques d'une organisation. Pour plus d'informations sur l'intégration des opérations cloud aux outils ITSM, veuillez consulter le guide d'intégration des opérations.

ITIL

Consultez la bibliothèque d'informations informatiques.

ITSM

Voir Gestion des services informatiques.

L

contrôle d'accès basé sur des étiquettes (LBAC)

Une implémentation du contrôle d'accès obligatoire (MAC) dans laquelle une valeur d'étiquette de sécurité est explicitement attribuée aux utilisateurs et aux données elles-mêmes. L'intersection entre l'étiquette de sécurité utilisateur et l'étiquette de sécurité des données détermine les lignes et les colonnes visibles par l'utilisateur.

zone de destination

Une zone d'atterrissage est un AWS environnement multi-comptes bien conçu, évolutif et sécurisé. Il s'agit d'un point de départ à partir duquel vos entreprises peuvent rapidement lancer et déployer des charges de travail et des applications en toute confiance dans leur environnement de sécurité et d'infrastructure. Pour plus d'informations sur les zones de destination, veuillez consulter Setting up a secure and scalable multi-account AWS environment.

grand modèle de langage (LLM)

Un modèle d'intelligence artificielle basé sur le deep learning qui est préentraîné sur une grande quantité de données. Un LLM peut effectuer plusieurs tâches, telles que répondre à des questions, résumer des documents, traduire du texte dans d'autres langues et compléter des phrases. Pour plus d'informations, voir Que sont LLMs.

migration de grande envergure

Migration de 300 serveurs ou plus.

LBAC

Voir contrôle d'accès basé sur des étiquettes.

principe de moindre privilège

Bonne pratique de sécurité qui consiste à accorder les autorisations minimales nécessaires à l'exécution d'une tâche. Pour plus d'informations, veuillez consulter la rubrique Accorder les autorisations de moindre privilège dans la documentation IAM.

lift and shift

Voir 7 Rs.

système de poids faible

Système qui stocke d'abord l'octet le moins significatif. Voir aussi endianité.

LLM

Voir le grand modèle de langage.

environnements inférieurs

Voir environnement.

M

machine learning (ML)

Type d'intelligence artificielle qui utilise des algorithmes et des techniques pour la reconnaissance et l'apprentissage de modèles. Le ML analyse et apprend à partir de données enregistrées, telles que les données de l'Internet des objets (IoT), pour générer un modèle statistique basé sur des modèles. Pour plus d'informations, veuillez consulter Machine Learning.

branche principale

Voir succursale.

malware

Logiciel conçu pour compromettre la sécurité ou la confidentialité de l'ordinateur. Les logiciels malveillants peuvent perturber les systèmes informatiques, divulguer des informations sensibles ou obtenir un accès non autorisé. Parmi les malwares, on peut citer les virus, les vers, les rançongiciels, les chevaux de Troie, les logiciels espions et les enregistreurs de frappe.

services gérés

Services AWS pour lequel AWS fonctionnent la couche d'infrastructure, le système d'exploitation et les plateformes, et vous accédez aux points de terminaison pour stocker et récupérer des données. Amazon Simple Storage Service (Amazon S3) et Amazon DynamoDB sont des exemples de services gérés. Ils sont également connus sous le nom de services abstraits.

système d'exécution de la fabrication (MES)

Un système logiciel pour le suivi, la surveillance, la documentation et le contrôle des processus de production qui convertissent les matières premières en produits finis dans l'atelier.

MAP

Voir Migration Acceleration Program.

mécanisme

Processus complet au cours duquel vous créez un outil, favorisez son adoption, puis inspectez les résultats afin de procéder aux ajustements nécessaires. Un mécanisme est un cycle qui se renforce et s'améliore lorsqu'il fonctionne. Pour plus d'informations, voir Création de mécanismes dans le cadre AWS Well-Architected.

compte membre

Tous, à l' Comptes AWS exception du compte de gestion, qui font partie d'une organisation dans AWS Organizations. Un compte ne peut être membre que d'une seule organisation à la fois.

MAILLES

Voir le système d'exécution de la fabrication.

Transport télémétrique en file d'attente de messages (MQTT)

Protocole de communication léger machine-to-machine (M2M), basé sur le modèle de publication/d'abonnement, pour les appareils IoT aux ressources limitées.

microservice

Un petit service indépendant qui communique via un réseau bien défini APIs et qui est généralement détenu par de petites équipes autonomes. Par exemple, un système d'assurance peut inclure des microservices qui mappent à des capacités métier, telles que les ventes ou le marketing, ou à des sous-domaines, tels que les achats, les réclamations ou l'analytique. Les avantages des microservices incluent l'agilité, la flexibilité de la mise à l'échelle, la facilité de déploiement, la réutilisation du code et la résilience. Pour plus d'informations, consultez la section Intégration de microservices à l'aide de services AWS sans serveur.

architecture de microservices

Approche de création d'une application avec des composants indépendants qui exécutent chaque processus d'application en tant que microservice. Ces microservices communiquent via une interface bien définie en utilisant Lightweight. APIs Chaque microservice de cette architecture peut être mis à jour, déployé et mis à l'échelle pour répondre à la demande de fonctions spécifiques d'une application. Pour plus d'informations, consultez la section Implémentation de microservices sur AWS.

Programme d'accélération des migrations (MAP)

Un AWS programme qui fournit un support de conseil, des formations et des services pour aider les entreprises à établir une base opérationnelle solide pour passer au cloud, et pour aider à compenser le coût initial des migrations. MAP inclut une méthodologie de migration pour exécuter les migrations héritées de manière méthodique, ainsi qu'un ensemble d'outils pour automatiser et accélérer les scénarios de migration courants.

migration à grande échelle

Processus consistant à transférer la majeure partie du portefeuille d'applications vers le cloud par vagues, un plus grand nombre d'applications étant déplacées plus rapidement à chaque vague. Cette phase utilise les bonnes pratiques et les enseignements tirés des phases précédentes pour implémenter une usine de migration d'équipes, d'outils et de processus en vue de rationaliser la migration des charges de travail grâce à l'automatisation et à la livraison agile. Il s'agit de la troisième phase de la stratégie de migration AWS.

usine de migration

Équipes interfonctionnelles qui rationalisent la migration des charges de travail grâce à des approches automatisées et agiles. Les équipes de Migration Factory comprennent généralement des responsables des opérations, des analystes commerciaux et des propriétaires, des ingénieurs de migration, des développeurs et DevOps des professionnels travaillant dans le cadre de sprints.

Entre 20 et 50 % du portefeuille d'applications d'entreprise est constitué de modèles répétés qui peuvent être optimisés par une approche d'usine. Pour plus d'informations, veuillez consulter la rubrique discussion of migration factories et le guide Cloud Migration Factory dans cet ensemble de contenus.

métadonnées de migration

Informations relatives à l'application et au serveur nécessaires pour finaliser la migration. Chaque modèle de migration nécessite un ensemble de métadonnées de migration différent. Les exemples de métadonnées de migration incluent le sous-réseau cible, le groupe de sécurité et le AWS compte.

modèle de migration

Tâche de migration reproductible qui détaille la stratégie de migration, la destination de la migration et l'application ou le service de migration utilisé. Exemple : réorganisez la migration vers Amazon EC2 AWS avec le service de migration d'applications.

Évaluation du portefeuille de migration (MPA)

Outil en ligne qui fournit des informations pour valider l'analyse de rentabilisation en faveur de la migration vers le. AWS Cloud La MPA propose une évaluation détaillée du portefeuille (dimensionnement approprié des serveurs, tarification, comparaison du coût total de possession, analyse des coûts de migration), ainsi que la planification de la migration (analyse et collecte des données d'applications, regroupement des applications, priorisation des migrations et planification des vagues). L'outil MPA (connexion requise) est disponible gratuitement pour tous les AWS consultants et consultants APN Partner.

Évaluation de la préparation à la migration (MRA)

Processus qui consiste à obtenir des informations sur l'état de préparation d'une organisation au cloud, à identifier les forces et les faiblesses et à élaborer un plan d'action pour combler les lacunes identifiées, à l'aide du AWS CAF. Pour plus d'informations, veuillez consulter le guide de préparation à la migration. La MRA est la première phase de la stratégie de migration AWS.

stratégie de migration

L'approche utilisée pour migrer une charge de travail vers le AWS Cloud. Pour plus d'informations, reportez-vous aux 7 R de ce glossaire et à Mobiliser votre organisation pour accélérer les migrations à grande échelle.

ML

Voir apprentissage automatique.

modernisation

Transformation d'une application obsolète (héritée ou monolithique) et de son infrastructure en un système agile, élastique et hautement disponible dans le cloud afin de réduire les coûts, de gagner en efficacité et de tirer parti des innovations. Pour plus d'informations, consultez la section Stratégie de modernisation des applications dans le AWS Cloud.

évaluation de la préparation à la modernisation

Évaluation qui permet de déterminer si les applications d'une organisation sont prêtes à être modernisées, d'identifier les avantages, les risques et les dépendances, et qui détermine dans quelle mesure l'organisation peut prendre en charge l'état futur de ces applications. Le résultat de l'évaluation est un plan de l'architecture cible, une feuille de route détaillant les phases de développement et les étapes du processus de modernisation, ainsi qu'un plan d'action pour combler les lacunes identifiées. Pour plus d'informations, consultez la section Évaluation de l'état de préparation à la modernisation des applications dans le AWS Cloud.

applications monolithiques (monolithes)

Applications qui s'exécutent en tant que service unique avec des processus étroitement couplés. Les applications monolithiques ont plusieurs inconvénients. Si une fonctionnalité de l'application connaît un pic de demande, l'architecture entière doit être mise à l'échelle. L'ajout ou l'amélioration des fonctionnalités d'une application monolithique devient également plus complexe lorsque la base de code s'élargit. Pour résoudre ces problèmes, vous pouvez utiliser une architecture de microservices. Pour plus d'informations, veuillez consulter Decomposing monoliths into microservices.

MPA

Voir Évaluation du portefeuille de migration.

MQTT

Voir Message Queuing Telemetry Transport.

classification multi-classes

Processus qui permet de générer des prédictions pour plusieurs classes (prédiction d'un résultat parmi plus de deux). Par exemple, un modèle de ML peut demander « Ce produit est-il un livre, une voiture ou un téléphone ? » ou « Quelle catégorie de produits intéresse le plus ce client ? ».

infrastructure mutable

Modèle qui met à jour et modifie l'infrastructure existante pour les charges de travail de production. Pour améliorer la cohérence, la fiabilité et la prévisibilité, le AWS Well-Architected Framework recommande l'utilisation d'une infrastructure immuable comme meilleure pratique.

O

OAC

Voir Contrôle d'accès à l'origine.

OAI

Voir l'identité d'accès à l'origine.

OCM

Voir gestion du changement organisationnel.

migration hors ligne

Méthode de migration dans laquelle la charge de travail source est supprimée au cours du processus de migration. Cette méthode implique un temps d'arrêt prolongé et est généralement utilisée pour de petites charges de travail non critiques.

OI

Consultez la section Intégration des opérations.

OLA

Voir l'accord au niveau opérationnel.

migration en ligne

Méthode de migration dans laquelle la charge de travail source est copiée sur le système cible sans être mise hors ligne. Les applications connectées à la charge de travail peuvent continuer à fonctionner pendant la migration. Cette méthode implique un temps d'arrêt nul ou minimal et est généralement utilisée pour les charges de travail de production critiques.

OPC-UA

Voir Open Process Communications - Architecture unifiée.

Communications par processus ouvert - Architecture unifiée (OPC-UA)

Un protocole de communication machine-to-machine (M2M) pour l'automatisation industrielle. L'OPC-UA fournit une norme d'interopérabilité avec des schémas de cryptage, d'authentification et d'autorisation des données.

accord au niveau opérationnel (OLA)

Accord qui précise ce que les groupes informatiques fonctionnels s'engagent à fournir les uns aux autres, afin de prendre en charge un contrat de niveau de service (SLA).

examen de l'état de préparation opérationnelle (ORR)

Une liste de questions et de bonnes pratiques associées qui vous aident à comprendre, à évaluer, à prévenir ou à réduire l'ampleur des incidents et des défaillances possibles. Pour plus d'informations, voir Operational Readiness Reviews (ORR) dans le AWS Well-Architected Framework.

technologie opérationnelle (OT)

Systèmes matériels et logiciels qui fonctionnent avec l'environnement physique pour contrôler les opérations, les équipements et les infrastructures industriels. Dans le secteur manufacturier, l'intégration des systèmes OT et des technologies de l'information (IT) est au cœur des transformations de l'industrie 4.0.

intégration des opérations (OI)

Processus de modernisation des opérations dans le cloud, qui implique la planification de la préparation, l'automatisation et l'intégration. Pour en savoir plus, veuillez consulter le guide d'intégration des opérations.

journal de suivi d'organisation

Un parcours créé par AWS CloudTrail qui enregistre tous les événements pour tous les membres Comptes AWS d'une organisation dans AWS Organizations. Ce journal de suivi est créé dans chaque Compte AWS qui fait partie de l'organisation et suit l'activité de chaque compte. Pour plus d'informations, consultez la section Création d'un suivi pour une organisation dans la CloudTrail documentation.

gestion du changement organisationnel (OCM)

Cadre pour gérer les transformations métier majeures et perturbatrices du point de vue des personnes, de la culture et du leadership. L'OCM aide les organisations à se préparer et à effectuer la transition vers de nouveaux systèmes et de nouvelles politiques en accélérant

l'adoption des changements, en abordant les problèmes de transition et en favorisant des changements culturels et organisationnels. Dans la stratégie de AWS migration, ce cadre est appelé accélération du personnel, en raison de la rapidité du changement requise dans les projets d'adoption du cloud. Pour plus d'informations, veuillez consulter le guide OCM.

contrôle d'accès d'origine (OAC)

Dans CloudFront, une option améliorée pour restreindre l'accès afin de sécuriser votre contenu Amazon Simple Storage Service (Amazon S3). L'OAC prend en charge tous les compartiments S3 dans leur ensemble Régions AWS, le chiffrement côté serveur avec AWS KMS (SSE-KMS) et les requêtes dynamiques PUT adressées au compartiment S3. DELETE

identité d'accès d'origine (OAI)

Dans CloudFront, une option permettant de restreindre l'accès afin de sécuriser votre contenu Amazon S3. Lorsque vous utilisez OAI, il CloudFront crée un principal auprès duquel Amazon S3 peut s'authentifier. Les principaux authentifiés peuvent accéder au contenu d'un compartiment S3 uniquement via une distribution spécifique CloudFront . Voir également OAC, qui fournit un contrôle d'accès plus précis et amélioré.

ORR

Voir l'examen de l'état de préparation opérationnelle.

DE

Voir technologie opérationnelle.

VPC sortant (de sortie)

Dans une architecture AWS multi-comptes, un VPC qui gère les connexions réseau initiées depuis une application. L'architecture AWS de référence de sécurité recommande de configurer votre compte réseau avec les fonctions entrantes, sortantes et d'inspection VPCs afin de protéger l'interface bidirectionnelle entre votre application et l'Internet en général.

P

limite des autorisations

Politique de gestion IAM attachée aux principaux IAM pour définir les autorisations maximales que peut avoir l'utilisateur ou le rôle. Pour plus d'informations, veuillez consulter la rubrique Limites des autorisations dans la documentation IAM.

informations personnelles identifiables (PII)

Informations qui, lorsqu'elles sont consultées directement ou associées à d'autres données connexes, peuvent être utilisées pour déduire raisonnablement l'identité d'une personne. Les exemples d'informations personnelles incluent les noms, les adresses et les informations de contact.

PII

Voir les informations personnelles identifiables.

manuel stratégique

Ensemble d'étapes prédéfinies qui capturent le travail associé aux migrations, comme la fourniture de fonctions d'opérations de base dans le cloud. Un manuel stratégique peut revêtir la forme de scripts, de runbooks automatisés ou d'un résumé des processus ou des étapes nécessaires au fonctionnement de votre environnement modernisé.

PLC

Voir contrôleur logique programmable.

PLM

Consultez la section Gestion du cycle de vie des produits.

policy

Objet capable de définir les autorisations (voir la politique basée sur l'identité), de spécifier les conditions d'accès (voir la politique basée sur les ressources) ou de définir les autorisations maximales pour tous les comptes d'une organisation dans AWS Organizations (voir la politique de contrôle des services).

persistance polyglotte

Choix indépendant de la technologie de stockage de données d'un microservice en fonction des modèles d'accès aux données et d'autres exigences. Si vos microservices utilisent la même technologie de stockage de données, ils peuvent rencontrer des difficultés d'implémentation ou présenter des performances médiocres. Les microservices sont plus faciles à mettre en œuvre, atteignent de meilleures performances, ainsi qu'une meilleure capacité de mise à l'échelle s'ils utilisent l'entrepôt de données le mieux adapté à leurs besoins.

évaluation du portefeuille

Processus de découverte, d'analyse et de priorisation du portefeuille d'applications afin de planifier la migration. Pour plus d'informations, veuillez consulter Evaluating migration readiness.

predicate

Une condition de requête qui renvoie `true` ou`false`, généralement située dans une `WHERE` clause.

prédicat pushdown

Technique d'optimisation des requêtes de base de données qui filtre les données de la requête avant le transfert. Cela réduit la quantité de données qui doivent être extraites et traitées à partir de la base de données relationnelle et améliore les performances des requêtes.

contrôle préventif

Contrôle de sécurité conçu pour empêcher qu'un événement ne se produise. Ces contrôles constituent une première ligne de défense pour empêcher tout accès non autorisé ou toute modification indésirable de votre réseau. Pour plus d'informations, veuillez consulter Preventative controls dans Implementing security controls on AWS.

principal

Entité AWS capable d'effectuer des actions et d'accéder aux ressources. Cette entité est généralement un utilisateur root pour un Compte AWS rôle IAM ou un utilisateur. Pour plus d'informations, veuillez consulter la rubrique Principal dans Termes et concepts relatifs aux rôles, dans la documentation IAM.

confidentialité dès la conception

Une approche d'ingénierie système qui prend en compte la confidentialité tout au long du processus de développement.

zones hébergées privées

Conteneur contenant des informations sur la manière dont vous souhaitez qu'Amazon Route 53 réponde aux requêtes DNS pour un domaine et ses sous-domaines au sein d'un ou de plusieurs VPCs domaines. Pour plus d'informations, veuillez consulter Working with private hosted zones dans la documentation Route 53.

contrôle proactif

Contrôle de sécurité conçu pour empêcher le déploiement de ressources non conformes. Ces contrôles analysent les ressources avant qu'elles ne soient provisionnées. Si la ressource n'est pas conforme au contrôle, elle n'est pas provisionnée. Pour plus d'informations, consultez le guide de référence sur les contrôles dans la AWS Control Tower documentation et consultez la section Contrôles proactifs dans Implémentation des contrôles de sécurité sur AWS.

gestion du cycle de vie des produits (PLM)

Gestion des données et des processus d'un produit tout au long de son cycle de vie, depuis la conception, le développement et le lancement, en passant par la croissance et la maturité, jusqu'au déclin et au retrait.

environnement de production

Voir environnement.

contrôleur logique programmable (PLC)

Dans le secteur manufacturier, un ordinateur hautement fiable et adaptable qui surveille les machines et automatise les processus de fabrication.

chaînage rapide

Utiliser le résultat d'une invite LLM comme entrée pour l'invite suivante afin de générer de meilleures réponses. Cette technique est utilisée pour décomposer une tâche complexe en sous-tâches ou pour affiner ou développer de manière itérative une réponse préliminaire. Cela permet d'améliorer la précision et la pertinence des réponses d'un modèle et permet d'obtenir des résultats plus précis et personnalisés.

pseudonymisation

Processus de remplacement des identifiants personnels dans un ensemble de données par des valeurs fictives. La pseudonymisation peut contribuer à protéger la vie privée. Les données pseudonymisées sont toujours considérées comme des données personnelles.

publish/subscribe (pub/sub)

Modèle qui permet les communications asynchrones entre les microservices afin d'améliorer l'évolutivité et la réactivité. Par exemple, dans un MES basé sur des microservices, un microservice peut publier des messages d'événements sur un canal auquel d'autres microservices peuvent s'abonner. Le système peut ajouter de nouveaux microservices sans modifier le service de publication.

Q

plan de requête

Série d'étapes, telles que des instructions, utilisées pour accéder aux données d'un système de base de données relationnelle SQL.

régression du plan de requêtes

Le cas où un optimiseur de service de base de données choisit un plan moins optimal qu'avant une modification donnée de l'environnement de base de données. Cela peut être dû à des changements en termes de statistiques, de contraintes, de paramètres d'environnement, de liaisons de paramètres de requêtes et de mises à jour du moteur de base de données.

R

Matrice RACI

Voir responsable, responsable, consulté, informé (RACI).

RAG

Voir Retrieval Augmented Generation.

rançongiciel

Logiciel malveillant conçu pour bloquer l'accès à un système informatique ou à des données jusqu'à ce qu'un paiement soit effectué.

Matrice RASCI

Voir responsable, responsable, consulté, informé (RACI).

RCAC

Voir contrôle d'accès aux lignes et aux colonnes.

réplica en lecture

Copie d'une base de données utilisée en lecture seule. Vous pouvez acheminer les requêtes vers le réplica de lecture pour réduire la charge sur votre base de données principale.

réarchitecte

Voir 7 Rs.

objectif de point de récupération (RPO)

Durée maximale acceptable depuis le dernier point de récupération des données. Il détermine ce qui est considéré comme étant une perte de données acceptable entre le dernier point de reprise et l’interruption du service.

objectif de temps de récupération (RTO)

Le délai maximum acceptable entre l'interruption du service et le rétablissement du service.

refactoriser

Voir 7 Rs.

Région

Un ensemble de AWS ressources dans une zone géographique. Chacun Région AWS est isolé et indépendant des autres pour garantir tolérance aux pannes, stabilité et résilience. Pour plus d'informations, voir Spécifier ce que Régions AWS votre compte peut utiliser.

régression

Technique de ML qui prédit une valeur numérique. Par exemple, pour résoudre le problème « Quel sera le prix de vente de cette maison ? », un modèle de ML pourrait utiliser un modèle de régression linéaire pour prédire le prix de vente d'une maison sur la base de faits connus à son sujet (par exemple, la superficie en mètres carrés).

réhéberger

Voir 7 Rs.

version

Dans un processus de déploiement, action visant à promouvoir les modifications apportées à un environnement de production.

déplacer

Voir 7 Rs.

replateforme

Voir 7 Rs.

rachat

Voir 7 Rs.

résilience

La capacité d'une application à résister aux perturbations ou à s'en remettre. La haute disponibilité et la reprise après sinistre sont des considérations courantes lors de la planification de la résilience dans le AWS Cloud. Pour plus d'informations, consultez AWS Cloud Résilience.

politique basée sur les ressources

Politique attachée à une ressource, comme un compartiment Amazon S3, un point de terminaison ou une clé de chiffrement. Ce type de politique précise les principaux auxquels l'accès est autorisé, les actions prises en charge et toutes les autres conditions qui doivent être remplies.

matrice responsable, redevable, consulté et informé (RACI)

Une matrice qui définit les rôles et les responsabilités de toutes les parties impliquées dans les activités de migration et les opérations cloud. Le nom de la matrice est dérivé des types de responsabilité définis dans la matrice : responsable (R), responsable (A), consulté (C) et informé (I). Le type de support (S) est facultatif. Si vous incluez le support, la matrice est appelée matrice RASCI, et si vous l'excluez, elle est appelée matrice RACI.

contrôle réactif

Contrôle de sécurité conçu pour permettre de remédier aux événements indésirables ou aux écarts par rapport à votre référence de sécurité. Pour plus d'informations, veuillez consulter la rubrique Responsive controls dans Implementing security controls on AWS.

retain

Voir 7 Rs.

se retirer

Voir 7 Rs.

Génération augmentée de récupération (RAG)

Technologie d'IA générative dans laquelle un LLM fait référence à une source de données faisant autorité qui se trouve en dehors de ses sources de données de formation avant de générer une réponse. Par exemple, un modèle RAG peut effectuer une recherche sémantique dans la base de connaissances ou dans les données personnalisées d'une organisation. Pour plus d'informations, voir Qu'est-ce que RAG ?

rotation

Processus de mise à jour périodique d'un secret pour empêcher un attaquant d'accéder aux informations d'identification.

contrôle d'accès aux lignes et aux colonnes (RCAC)

Utilisation d'expressions SQL simples et flexibles dotées de règles d'accès définies. Le RCAC comprend des autorisations de ligne et des masques de colonnes.

RPO

Voir l'objectif du point de récupération.

RTO

Voir l'objectif en matière de temps de rétablissement.

runbook

Ensemble de procédures manuelles ou automatisées nécessaires à l'exécution d'une tâche spécifique. Elles visent généralement à rationaliser les opérations ou les procédures répétitives présentant des taux d'erreur élevés.

S

SAML 2.0

Un standard ouvert utilisé par de nombreux fournisseurs d'identité (IdPs). Cette fonctionnalité permet l'authentification unique fédérée (SSO), afin que les utilisateurs puissent se connecter AWS Management Console ou appeler les opérations de l' AWS API sans que vous ayez à créer un utilisateur dans IAM pour tous les membres de votre organisation. Pour plus d'informations sur la fédération SAML 2.0, veuillez consulter À propos de la fédération SAML 2.0 dans la documentation IAM.

SCADA

Voir Contrôle de supervision et acquisition de données.

SCP

Voir la politique de contrôle des services.

secret

Dans AWS Secrets Manager des informations confidentielles ou restreintes, telles qu'un mot de passe ou des informations d'identification utilisateur, que vous stockez sous forme cryptée. Il comprend la valeur secrète et ses métadonnées. La valeur secrète peut être binaire, une chaîne unique ou plusieurs chaînes. Pour plus d'informations, voir Que contient le secret d'un Secrets Manager ? dans la documentation de Secrets Manager.

sécurité dès la conception

Une approche d'ingénierie système qui prend en compte la sécurité tout au long du processus de développement.

contrôle de sécurité

Barrière de protection technique ou administrative qui empêche, détecte ou réduit la capacité d'un assaillant d'exploiter une vulnérabilité de sécurité. Il existe quatre principaux types de contrôles de sécurité : préventifs, détectifs, réactifs et proactifs.

renforcement de la sécurité

Processus qui consiste à réduire la surface d'attaque pour la rendre plus résistante aux attaques. Cela peut inclure des actions telles que la suppression de ressources qui ne sont plus requises, la mise en œuvre des bonnes pratiques de sécurité consistant à accorder le moindre privilège ou la désactivation de fonctionnalités inutiles dans les fichiers de configuration.

système de gestion des informations et des événements de sécurité (SIEM)

Outils et services qui associent les systèmes de gestion des informations de sécurité (SIM) et de gestion des événements de sécurité (SEM). Un système SIEM collecte, surveille et analyse les données provenant de serveurs, de réseaux, d'appareils et d'autres sources afin de détecter les menaces et les failles de sécurité, mais aussi de générer des alertes.

automatisation des réponses de sécurité

Action prédéfinie et programmée conçue pour répondre automatiquement à un événement de sécurité ou y remédier. Ces automatisations servent de contrôles de sécurité détectifs ou réactifs qui vous aident à mettre en œuvre les meilleures pratiques en matière AWS de sécurité. Parmi les actions de réponse automatique, citons la modification d'un groupe de sécurité VPC, l'application de correctifs à une instance Amazon EC2 ou la rotation des informations d'identification.

chiffrement côté serveur

Chiffrement des données à destination, par celui Service AWS qui les reçoit.

Politique de contrôle des services (SCP)

Politique qui fournit un contrôle centralisé des autorisations pour tous les comptes d'une organisation dans AWS Organizations. SCPs définissez des garde-fous ou des limites aux actions qu'un administrateur peut déléguer à des utilisateurs ou à des rôles. Vous pouvez les utiliser SCPs comme listes d'autorisation ou de refus pour spécifier les services ou les actions autorisés ou interdits. Pour plus d'informations, consultez la section Politiques de contrôle des services dans la AWS Organizations documentation.

point de terminaison du service

URL du point d'entrée pour un Service AWS. Pour vous connecter par programmation au service cible, vous pouvez utiliser un point de terminaison. Pour plus d'informations, veuillez consulter la rubrique Service AWS endpoints dans Références générales AWS.

contrat de niveau de service (SLA)

Accord qui précise ce qu'une équipe informatique promet de fournir à ses clients, comme le temps de disponibilité et les performances des services.

indicateur de niveau de service (SLI)

Mesure d'un aspect des performances d'un service, tel que son taux d'erreur, sa disponibilité ou son débit.

objectif de niveau de service (SLO)

Mesure cible qui représente l'état d'un service, tel que mesuré par un indicateur de niveau de service.

modèle de responsabilité partagée

Un modèle décrivant la responsabilité que vous partagez en matière AWS de sécurité et de conformité dans le cloud. AWS est responsable de la sécurité du cloud, alors que vous êtes responsable de la sécurité dans le cloud. Pour de plus amples informations, veuillez consulter Modèle de responsabilité partagée.

SIEM

Consultez les informations de sécurité et le système de gestion des événements.

point de défaillance unique (SPOF)

Défaillance d'un seul composant critique d'une application susceptible de perturber le système.

SLA

Voir le contrat de niveau de service.

SLI

Voir l'indicateur de niveau de service.

SLO

Voir l'objectif de niveau de service.

split-and-seed modèle

Modèle permettant de mettre à l'échelle et d'accélérer les projets de modernisation. Au fur et à mesure que les nouvelles fonctionnalités et les nouvelles versions de produits sont définies, l'équipe principale se divise pour créer des équipes de produit. Cela permet de mettre à l'échelle les capacités et les services de votre organisation, d'améliorer la productivité des développeurs et de favoriser une innovation rapide. Pour plus d'informations, voir Approche progressive de la modernisation des applications dans le. AWS Cloud

SPOF

Voir point de défaillance unique.

schéma en étoile

Structure organisationnelle de base de données qui utilise une grande table de faits pour stocker les données transactionnelles ou mesurées et utilise une ou plusieurs tables dimensionnelles plus petites pour stocker les attributs des données. Cette structure est conçue pour être utilisée dans un entrepôt de données ou à des fins de business intelligence.

modèle de figuier étrangleur

Approche de modernisation des systèmes monolithiques en réécrivant et en remplaçant progressivement les fonctionnalités du système jusqu'à ce que le système hérité puisse être mis hors service. Ce modèle utilise l'analogie d'un figuier de vigne qui se développe dans un arbre existant et qui finit par supplanter son hôte. Le schéma a été présenté par Martin Fowler comme un moyen de gérer les risques lors de la réécriture de systèmes monolithiques. Pour obtenir un exemple d'application de ce modèle, veuillez consulter Modernizing legacy Microsoft ASP.NET (ASMX) web services incrementally by using containers and Amazon API Gateway.

sous-réseau

Plage d'adresses IP dans votre VPC. Un sous-réseau doit se trouver dans une seule zone de disponibilité.

contrôle de supervision et acquisition de données (SCADA)

Dans le secteur manufacturier, un système qui utilise du matériel et des logiciels pour surveiller les actifs physiques et les opérations de production.

chiffrement symétrique

Algorithme de chiffrement qui utilise la même clé pour chiffrer et déchiffrer les données.

tests synthétiques

Tester un système de manière à simuler les interactions des utilisateurs afin de détecter les problèmes potentiels ou de surveiller les performances. Vous pouvez utiliser Amazon CloudWatch Synthetics pour créer ces tests.

invite du système

Technique permettant de fournir un contexte, des instructions ou des directives à un LLM afin d'orienter son comportement. Les instructions du système aident à définir le contexte et à établir des règles pour les interactions avec les utilisateurs.

T

tags

Des paires clé-valeur qui agissent comme des métadonnées pour organiser vos AWS ressources. Les balises peuvent vous aider à gérer, identifier, organiser, rechercher et filtrer des ressources. Pour plus d'informations, veuillez consulter la rubrique Balisage de vos AWS ressources.

variable cible

La valeur que vous essayez de prédire dans le cadre du ML supervisé. Elle est également qualifiée de variable de résultat. Par exemple, dans un environnement de fabrication, la variable cible peut être un défaut du produit.

liste de tâches

Outil utilisé pour suivre les progrès dans un runbook. Liste de tâches qui contient une vue d'ensemble du runbook et une liste des tâches générales à effectuer. Pour chaque tâche générale, elle inclut le temps estimé nécessaire, le propriétaire et l'avancement.

environnement de test

Voir environnement.

entraînement

Pour fournir des données à partir desquelles votre modèle de ML peut apprendre. Les données d'entraînement doivent contenir la bonne réponse. L'algorithme d'apprentissage identifie des modèles dans les données d'entraînement, qui mettent en correspondance les attributs des données d'entrée avec la cible (la réponse que vous souhaitez prédire). Il fournit un modèle de ML

qui capture ces modèles. Vous pouvez alors utiliser le modèle de ML pour obtenir des prédictions sur de nouvelles données pour lesquelles vous ne connaissez pas la cible.

passerelle de transit

Un hub de transit réseau que vous pouvez utiliser pour interconnecter vos réseaux VPCs et ceux sur site. Pour plus d'informations, voir Qu'est-ce qu'une passerelle de transit dans la AWS Transit Gateway documentation.

flux de travail basé sur jonction

Approche selon laquelle les développeurs génèrent et testent des fonctionnalités localement dans une branche de fonctionnalités, puis fusionnent ces modifications dans la branche principale. La branche principale est ensuite intégrée aux environnements de développement, de préproduction et de production, de manière séquentielle.

accès sécurisé

Accorder des autorisations à un service que vous spécifiez pour effectuer des tâches au sein de votre organisation AWS Organizations et dans ses comptes en votre nom. Le service de confiance crée un rôle lié au service dans chaque compte, lorsque ce rôle est nécessaire, pour effectuer des tâches de gestion à votre place. Pour plus d'informations, consultez la section Utilisation AWS Organizations avec d'autres AWS services dans la AWS Organizations documentation.

réglage

Pour modifier certains aspects de votre processus d'entraînement afin d'améliorer la précision du modèle de ML. Par exemple, vous pouvez entraîner le modèle de ML en générant un ensemble d'étiquetage, en ajoutant des étiquettes, puis en répétant ces étapes plusieurs fois avec différents paramètres pour optimiser le modèle.

équipe de deux pizzas

Une petite DevOps équipe que vous pouvez nourrir avec deux pizzas. Une équipe de deux pizzas garantit les meilleures opportunités de collaboration possible dans le développement de logiciels.

U

incertitude

Un concept qui fait référence à des informations imprécises, incomplètes ou inconnues susceptibles de compromettre la fiabilité des modèles de ML prédictifs. Il existe deux types

d'incertitude : l'incertitude épistémique est causée par des données limitées et incomplètes, alors que l'incertitude aléatoire est causée par le bruit et le caractère aléatoire inhérents aux données. Pour plus d'informations, veuillez consulter le guide Quantifying uncertainty in deep learning systems.

tâches indifférenciées

Également connu sous le nom de « levage de charges lourdes », ce travail est nécessaire pour créer et exploiter une application, mais qui n'apporte pas de valeur directe à l'utilisateur final ni d'avantage concurrentiel. Les exemples de tâches indifférenciées incluent l'approvisionnement, la maintenance et la planification des capacités.

environnements supérieurs

Voir environnement.

V

mise à vide

Opération de maintenance de base de données qui implique un nettoyage après des mises à jour incrémentielles afin de récupérer de l'espace de stockage et d'améliorer les performances.

contrôle de version

Processus et outils permettant de suivre les modifications, telles que les modifications apportées au code source dans un référentiel.

Appairage de VPC

Une connexion entre deux VPCs qui vous permet d'acheminer le trafic en utilisant des adresses IP privées. Pour plus d'informations, veuillez consulter la rubrique Qu'est-ce que l'appairage de VPC ? dans la documentation Amazon VPC.

vulnérabilités

Défaut logiciel ou matériel qui compromet la sécurité du système.

W

cache actif

Cache tampon qui contient les données actuelles et pertinentes fréquemment consultées. L'instance de base de données peut lire à partir du cache tampon, ce qui est plus rapide que la lecture à partir de la mémoire principale ou du disque.

données chaudes

Données rarement consultées. Lorsque vous interrogez ce type de données, des requêtes modérément lentes sont généralement acceptables.

fonction de fenêtre

Fonction SQL qui effectue un calcul sur un groupe de lignes liées d'une manière ou d'une autre à l'enregistrement en cours. Les fonctions de fenêtre sont utiles pour traiter des tâches, telles que le calcul d'une moyenne mobile ou l'accès à la valeur des lignes en fonction de la position relative de la ligne en cours.

charge de travail

Ensemble de ressources et de code qui fournit une valeur métier, par exemple une application destinée au client ou un processus de backend.

flux de travail

Groupes fonctionnels d'un projet de migration chargés d'un ensemble de tâches spécifique. Chaque flux de travail est indépendant, mais prend en charge les autres flux de travail du projet. Par exemple, le flux de travail du portefeuille est chargé de prioriser les applications, de planifier les vagues et de collecter les métadonnées de migration. Le flux de travail du portefeuille fournit ces actifs au flux de travail de migration, qui migre ensuite les serveurs et les applications.

VER

Voir écrire une fois, lire plusieurs.

WQF

Voir le cadre AWS de qualification de la charge de travail.

écrire une fois, lire plusieurs (WORM)

Modèle de stockage qui écrit les données une seule fois et empêche leur suppression ou leur modification. Les utilisateurs autorisés peuvent lire les données autant de fois que nécessaire,

mais ils ne peuvent pas les modifier. Cette infrastructure de stockage de données est considérée comme immuable.

Z

exploit Zero-Day

Une attaque, généralement un logiciel malveillant, qui tire parti d'une vulnérabilité de type « jour zéro ».

vulnérabilité « jour zéro »

Une faille ou une vulnérabilité non atténuée dans un système de production. Les acteurs malveillants peuvent utiliser ce type de vulnérabilité pour attaquer le système. Les développeurs prennent souvent conscience de la vulnérabilité à la suite de l'attaque.

invite Zero-Shot

Fournir à un LLM des instructions pour effectuer une tâche, mais aucun exemple (plans) pouvant aider à la guider. Le LLM doit utiliser ses connaissances pré-entraînées pour gérer la tâche. L'efficacité de l'invite zéro dépend de la complexité de la tâche et de la qualité de l'invite. Voir également les instructions en quelques clics.

application zombie

Application dont l'utilisation moyenne du processeur et de la mémoire est inférieure à 5 %. Dans un projet de migration, il est courant de retirer ces applications.

Les traductions sont fournies par des outils de traduction automatique. En cas de conflit entre le contenu d'une traduction et celui de la version originale en anglais, la version anglaise prévaudra.