Anwenden des AWS Well-Architected Frameworks für Amazon Neptune

AWS Prescriptive Guidance

AWS Prescriptive Guidance: Anwenden des AWS Well-Architected Frameworks für Amazon Neptune

Table of Contents

Anwendung des AWS Well-Architected Frameworks für Amazon Neptune

Amazon Web Services (Mitwirkende)

Januar 2026 (Verlauf der Dokumente)

Sie können graphbasierte Lösungen auf Amazon Web Services (AWS) mithilfe von Amazon Neptune erstellen. Dieser Leitfaden enthält Anleitungen zur Anwendung der Prinzipien des AWS Well-Architected Framework bei der Planung Ihrer Neptune-Bereitstellung.

Das AWS Well-Architected Framework unterstützt Sie beim Aufbau sicherer, leistungsstarker, belastbarer und effizienter Infrastrukturen für eine Vielzahl von Anwendungen und Workloads. Es bietet Ihnen auch einen konsistenten Ansatz zur Bewertung von Architekturen und zur Implementierung skalierbarer Designs.

Das AWS Well-Architected Framework basiert auf den folgenden sechs Säulen:

- Operative Exzellenz
- Sicherheit
- Zuverlässigkeit
- Leistungseffizienz
- Kostenoptimierung
- Nachhaltigkeit

Dieses Handbuch enthält Informationen zu den Grundpfeilern und bewährten Methoden des AWS Well-Architected Framework-Designs sowie Überlegungen, die Sie bei der Bereitstellung von Neptune berücksichtigen sollten. AWS

Zielgruppe

Dieser Leitfaden richtet sich an Dateningenieure, Lösungsarchitekten und Datenanalysten, die Lösungen entwerfen und implementieren, bei denen Grafiken verwendet werden. AWS

Ziele

Dieser Leitfaden kann Ihnen und Ihrer Organisation dabei helfen, Folgendes zu tun:

- Wählen Sie je nach Anwendungsfall und Abfragemustern aus den unterstützten Bereitstellungsoptionen und Abfragesprachen.
- Folgen Sie den AWS Well-Architected-Entwurfsmustern, die zur Verbesserung der Widerstandsfähigkeit und Sicherheit beitragen.
- Entwerfen Sie Ihre Abfragen im Hinblick auf optimale Leistung und Kosteneinsparungen.
- Erfahren Sie, wie Sie Ihren Neptune-Cluster in der Produktion betrieblich effizient verwalten können.

Säule „Operational Excellence“

Die Säule Operational Excellence des AWS Well-Architected Framework konzentriert sich auf den Betrieb und die Überwachung von Systemen sowie die kontinuierliche Verbesserung von Prozessen und Verfahren. Dazu gehört die Fähigkeit, die Entwicklung zu unterstützen und Workloads effektiv auszuführen, Einblicke in deren Betrieb zu gewinnen und die unterstützenden Prozesse und Verfahren kontinuierlich zu verbessern, um einen Mehrwert für das Unternehmen zu erzielen. Sie können die betriebliche Komplexität reduzieren, indem Sie Workloads automatisch reparieren, wodurch die meisten Probleme ohne menschliches Eingreifen erkannt und behoben werden. Sie können auf dieses Ziel hinarbeiten, indem Sie die in diesem Abschnitt beschriebenen bewährten Methoden befolgen. Verwenden Sie die Kennzahlen und Mechanismen von Amazon Neptune APIs, um angemessen zu reagieren, wenn Ihre Arbeitslast vom erwarteten Verhalten abweicht.

Diese Diskussion über den Pfeiler Operational Excellence konzentriert sich auf die folgenden Schlüsselbereiche:

- Infrastructure as Code (IaC)
- Änderungsmanagement
- Strategien zur Resilienz
- Vorfallmanagement
- Auditberichte zur Einhaltung der Vorschriften
- Protokollierung und Überwachung

Automatisieren Sie die Bereitstellung mithilfe eines IaC-Ansatzes

Zu den bewährten Methoden für die Automatisierung der Bereitstellung auf Neptune mithilfe von IaC gehören:

- Wenden Sie nach Möglichkeit Infrastructure as Code (IaC) an, um Neptun-Cluster bereitzustellen. Verwenden Sie für eine konsistente Umgebungskonfiguration eine AWS CloudFormationVorlage oder HashiCorp Terraform AWS Cloud Development Kit (AWS CDK), um alle erforderlichen Ressourcen für Ihren Cluster zu erstellen.
- Automatisieren Sie Betriebsabläufe in Neptune, wie z. B. die Größenänderung von Instances, das Hinzufügen oder Entfernen von Read Replicas oder das Durchführen manueller Failovers für globale Tabellen, wann immer dies möglich ist.

- Speichern Sie Verbindungszeichenfolgen extern von Ihrem Client aus. Verwenden Sie ETL-Prozesse (Extrahieren, Transformieren und Laden), um blue/green Bereitstellungsstrategien, Disaster Recovery (DR) und Migrationen zu neuen Clustern nahezu ohne Ausfallzeiten zu vereinfachen. Verbindungszeichenfolgen können in AWS Secrets ManagerAmazon DynamoDB oder an einem beliebigen Ort gespeichert werden, wo sie dynamisch geändert werden können.
- Verwenden Sie Tags, um Metadaten zu Ihren Neptune-Ressourcen hinzuzufügen, und verfolgen Sie die Nutzung anhand von Tags. Weitere Informationen finden Sie unter Tagging Amazon Neptune Resources.

Nehmen Sie häufig kleine, umkehrbare Änderungen vor

Die folgenden Empfehlungen konzentrieren sich auf kleine, umkehrbare Änderungen, um die Komplexität zu minimieren und die Wahrscheinlichkeit einer Unterbrechung der Arbeitslast zu verringern:

- Speichern Sie IaC-Vorlagen und -Skripts in einem Quellcodeverwaltungsdienst, z. B. GitHub oder GitLab.

> **Important**
>
> Speichern Sie keine AWS Anmeldeinformationen in der Quellcodeverwaltung.

- Erfordern Sie, dass IaC-Bereitstellungen einen CI/CD-Dienst (Continuous Integration and Continuous Delivery) verwenden, z. B. oder. AWS CodePipelineAWS CodeBuild Diese Services kompilieren, testen und implementieren Code in einer Nicht-Produktionsumgebung, die einen kurzlebigen Neptune-Cluster enthält, bevor sie sich auf Ihren Amazon Neptune Neptune-Produktionscluster auswirken.
- Testen Sie Infrastruktur- und Anwendungsabfragen in einer niedrigeren Umgebung, bevor Sie sie in der Produktion einsetzen. Auf diese Weise wird die Wahrscheinlichkeit einer Unterbrechung minimiert und es wird sichergestellt, dass sie Ihrer Arbeitslast und Ihrem Umfang entsprechend funktionieren.

Rechnen Sie mit Ausfällen

Eine selbstreparierende Infrastruktur ist ein Beispiel für betriebliche Exzellenz, da sie Ausfälle antizipiert und versucht, Probleme ohne Eingreifen zu lösen. Die folgenden Empfehlungen helfen Ihnen dabei, diese Reife mit Neptune zu erreichen:

- Erstellen Sie einen Überwachungsplan, der CloudWatch Amazon-Metriken verwendet, um die CPU- und Speicherauslastung Ihrer DB-Instance zu überwachen und die Nutzungsmuster zu verstehen. Erstellen Sie CloudWatch Dashboards und Alarme für wichtige Kennzahlen und die Antworten der Neptune-Kunden in Ihren Anwendungsprotokollen. Weitere Informationen zu Indikatoren für hohe oder niedrige CPU-Auslastung finden Sie unter Verwendung CloudWatch zur Überwachung der DB-Instance-Leistung in Neptune in der Neptune-Dokumentation.

  Wenn bei Ihren Abfragen häufig out-of-memory Ausnahmen auftreten, sollten Sie die Gesamtzahl der Knoten reduzieren, die Ihre Abfrage durchläuft, oder versuchen Sie, eine Instance aus der X2 Familie zu verwenden, die ein höheres Verhältnis aufweist. RAM-to-CPU
- Richten Sie Benachrichtigungen ein, um den Zustand des Neptun-Clusters zu überwachen. Sie `BufferCacheHitRatio` sollte beispielsweise konstant hoch sein (über 99,9 Prozent), während sie konstant niedrig sein `MainRequestQueuePendingRequests` sollte (idealerweise 0, aber abhängig von Ihren Anforderungen und der Latenztoleranz).
- Erwägen Sie die Verwendung von Read Replicas, um eine hohe Verfügbarkeit innerhalb von Neptune zu erreichen. Sie sollten mindestens zwei Read Replicas in unterschiedlichen Availability Zones als die Writer-Instance haben, um sicherzustellen, dass während eines Failover-Ereignisses immer eine Instanz für Leseanfragen verfügbar ist.
- Automatische Skalierung von Read Replicas auf der Grundlage von Nutzungsmetriken. Weitere Informationen finden Sie unter Automatische Skalierung der Anzahl von Replikaten in einem Amazon Neptune Neptune-DB-Cluster.
- Testen Sie den Failover für Ihre DB-Instance, um zu verstehen, wie lange der Vorgang für Ihren Anwendungsfall dauert.
- Wenn Ihre Anwendung einen kompletten AWS-Region Ausfall überstehen muss, sollten Sie die Verwendung globaler Datenbanken als Teil Ihrer DR-Pläne in Betracht ziehen.

Lernen Sie aus allen Betriebsausfällen

Eine Infrastruktur zur Selbstheilung ist ein langfristiges Projekt, das sich in mehreren Schritten entwickelt, wenn seltene Probleme auftreten oder die Reaktionen nicht so effektiv sind wie

gewünscht. Durch die Anwendung der folgenden Methoden wird die Konzentration auf dieses Ziel vorangetrieben:

- Treiben Sie Verbesserungen voran, indem Sie aus allen Fehlern lernen.
- Teilen Sie das Gelernte mit den Teams und der Organisation. Wenn mehrere Teams innerhalb einer Organisation Neptune verwenden, richten Sie einen gemeinsamen Chatroom oder eine Benutzergruppe ein, um Erfahrungen und bewährte Verfahren auszutauschen.

Verwenden Sie Protokollierungsfunktionen, um unbefugte oder ungewöhnliche Aktivitäten zu überwachen

Um ungewöhnliche Leistungs- und Aktivitätsmuster zu beobachten, speichern Sie Protokolle in Amazon CloudWatch Logs. Bedenken Sie die folgenden bewährten Methoden:

- Aktivieren Sie die Protokollierung langsamer Abfragen. Überprüfen Sie regelmäßig das Protokoll und stellen Sie fest, warum bestimmte Abfragen langsam sind. Verwenden Sie Neptune Explain and Profile Endpoints für Gremlin, SPARQL oder OpenCypher, um zu verstehen, warum diese Abfragen langsam sind.
- Aktivieren Sie die Neptune-Auditprotokolle und überprüfen Sie die Protokolle regelmäßig auf unbefugten Zugriff oder Anomalien.
- Wenn Sie die Protokollierung mit langsamen Abfragen oder die Auditprotokollierung verwenden, aktivieren Sie die Veröffentlichung in Logs. CloudWatch Auf diese Weise können Sie vermeiden, dass Ihnen der Festplattenspeicher auf den Instanzen ausgeht. Neptune-Instances haben eine begrenzte Protokollspeicherkapazität und überschreiben ältere Protokolldateien, wenn der Protokollspeicher überschritten wird. CloudWatch Logs unterstützt die langfristige Aufbewahrung von Protokollen. Die erweiterten Überwachungsfunktionen in CloudWatch Logs verbessern Ihre Fähigkeit, Protokolle abzufragen und Probleme zu diagnostizieren.
- Um bessere Analysetools für Ihre Audit-Logs zu ermöglichen, können Sie einen Neptune-DB-Cluster so konfigurieren, dass er Audit-Log-Daten in einer Protokollgruppe in CloudWatch Logs veröffentlicht. Mit CloudWatch Logs können Sie die Protokolldaten in Echtzeit analysieren, Alarme erstellen und Metriken anzeigen und CloudWatch Logs verwenden, um Ihre Protokolldatensätze auf einem äußerst dauerhaften Speicher zu speichern. CloudWatch Weitere Informationen finden Sie unter Neptune-Protokolle in Amazon CloudWatch Logs veröffentlichen.

- Neptune unterstützt die Protokollierung von Aktionen auf der Kontrollebene mithilfe von. AWS CloudTrail Weitere Informationen finden Sie unter Protokollieren von Amazon Neptune Neptune-API-Aufrufen mit. AWS CloudTrail

Säule der Sicherheit

Cloud-Sicherheit hat AWS höchste Priorität. Als AWS Kunde profitieren Sie von einer Rechenzentrums- und Netzwerkarchitektur, die darauf ausgelegt sind, die Anforderungen der sicherheitssensibelsten Unternehmen zu erfüllen.

Sicherheit ist eine gemeinsame Verantwortung von Ihnen AWS und Ihnen. Im Modell der übergreifenden Verantwortlichkeit wird Folgendes mit „Sicherheit der Cloud“ bzw. „Sicherheit in der Cloud“ umschrieben:

- Sicherheit der Cloud — AWS ist verantwortlich für den Schutz der Infrastruktur, die AWS-Services in der läuft AWS Cloud. AWS bietet Ihnen auch Dienste, die Sie sicher nutzen können. Externe Prüfer testen und verifizieren regelmäßig die Wirksamkeit der AWS Sicherheit im Rahmen der AWS Compliance-Programme. Informationen zu den für Amazon Neptune geltenden Compliance-Programmen finden Sie unter Im Rahmen des Compliance-Programms zugelassene -Services.
- Sicherheit in der Cloud — Ihre Verantwortung richtet sich nach dem AWS-Service , was Sie verwenden. Sie sind auch für andere Faktoren verantwortlich, etwa für die Vertraulichkeit Ihrer Daten, die Anforderungen Ihres Unternehmens und die geltenden Gesetze und Vorschriften. Weitere Informationen zum Datenschutz finden Sie unter Häufig gestellte Fragen zum Datenschutz. Informationen zum Datenschutz in Europa finden Sie im Blogbeitrag AWS Shared Responsibility Model und GDPR.

Die Sicherheitssäule hilft Ihnen zu verstehen, wie Sie das Modell der gemeinsamen Verantwortung bei der Verwendung von Neptune anwenden können. Die folgenden Themen zeigen Ihnen, wie Sie Neptune zur Erfüllung Ihrer Sicherheits- und Compliance-Ziele konfigurieren können. Sie lernen auch, wie Sie andere verwenden können AWS-Services , die Ihnen helfen, Ihre Neptun-Ressourcen zu überwachen und zu sichern.

Die Sicherheitssäule umfasst die folgenden Schwerpunktbereiche:

- Datensicherheit
- Netzwerksicherheit
- Authentifizierung und Autorisierung

Implementieren Sie Datensicherheit

Datenlecks und Sicherheitsverletzungen gefährden Ihre Kunden und können erhebliche negative Auswirkungen auf Ihr Unternehmen haben. Die folgenden bewährten Methoden tragen dazu bei, Ihre Kundendaten vor unbeabsichtigter und böswilliger Offenlegung zu schützen:

- Clusternamen, Tags, Parametergruppen, AWS Identity and Access Management (IAM) -Rollen und andere Metadaten sollten keine vertraulichen oder sensiblen Informationen enthalten, da diese Daten in Abrechnungs- oder Diagnoseprotokollen erscheinen könnten.
- URIs oder Links zu externen Servern, die als Daten in Neptune gespeichert sind, sollten keine Anmeldeinformationen zur Validierung von Anfragen enthalten.
- Verschlüsselte Neptune-Instances bieten eine zusätzliche Datenschutzebene, da Sie Ihre Daten vor nicht autorisierten Zugriffen auf den zugrunde liegenden Speicher schützen. Sie können mithilfe der Neptune-Verschlüsselung den Datenschutz für Ihre in der Cloud bereitgestellten Anwendungen erhöhen. Sie können auch Neptune-Verschlüsselung verwenden, um die Compliance-Anforderungen für ruhende Daten zu erfüllen.

  Um die Verschlüsselung für eine neue Neptune-DB-Instance zu aktivieren, wählen Sie in der Neptune-Konsole im Abschnitt Verschlüsselung aktivieren die Option Ja aus (standardmäßig ausgewählt), oder legen Sie die Eigenschaft unter fest. AWS::Neptune::DBCluster::StorageEncrypted CloudFormation Wenn die Verschlüsselung aktiviert ist, verwendet Neptune standardmäßig den von Amazon Relational Database Service (Amazon RDS) verwalteten AWS-Schlüssel, oder Sie können einen vom Kunden verwalteten Schlüssel erstellen. Informationen zum Erstellen einer Neptune-DB-Instance finden Sie unter Erstellen eines neuen Neptune-DB-Clusters. Weitere Informationen finden Sie unter Neptune-Ressourcen im Ruhezustand verschlüsseln. Ihre automatisierten und manuellen Snapshots verwenden dieselbe Verschlüsselung, die Sie für Ihren Neptune-Cluster ausgewählt haben.
- Wenn Sie die Sprachen SPARQL und OpenCypher verwenden, sollten Sie die richtigen Techniken zur Eingabevalidierung und Parametrisierung anwenden, um SQL-Injection und andere Formen von Angriffen zu verhindern. Vermeiden Sie es, Abfragen zu erstellen, die eine Verkettung von Zeichenketten mit vom Benutzer bereitgestellten Eingaben verwenden. Verwenden Sie parametrisierte Abfragen oder vorbereitete Anweisungen, um Eingabeparameter sicher an die Graphdatenbank zu übergeben. Weitere Informationen finden Sie unter Beispiele für parametrisierte OpenCypher-Abfragen und SPARQL Injection Defence.
- Verwenden Sie für die Gremlin-Sprache Gremlin-Sprachvarianten, anstatt direkt auf Zeichenketten basierende Gremlin-Skripte zu übergeben, um mögliche Injektionsprobleme zu vermeiden.

Schützen Sie Ihre Netzwerke

Ein Amazon Neptune Neptune-DB-Cluster kann nur in einer Virtual Private Cloud (VPC) erstellt werden. AWS Bis Neptune 1.4.6.0 waren die Endpunkte des Neptune-DB-Clusters nur innerhalb dieser VPC zugänglich. Ab Neptune 1.4.6.0 und höher können Neptun-Instanzen so konfiguriert werden, dass sie über das Internet öffentlich zugänglich sind. Es hat sich bewährt, diese Funktion nur in Produktionsumgebungen zu verwenden, um Ihren Entwicklern einen vereinfachten Zugriff auf Neptune zu ermöglichen (allerdings ist für den öffentlichen Zugriff immer eine IAM-Authentifizierung erforderlich). Wenn Sie öffentlichen Zugriff aktiviert haben, sollten Sie erwägen, Sicherheitsgruppenregeln für eingehenden Datenverkehr für Ihren Datenbankport so einzurichten, dass nur bekannter IP-Adressverkehr verwendet wird. Schützen Sie Ihre Neptune-Daten in Produktionsumgebungen oder mit Clustern, die vertrauliche Daten enthalten, indem Sie den öffentlichen Zugriff verhindern und den Zugriff auf die VPC einschränken, in der sich Ihr Neptune-DB-Cluster befindet. Weitere Informationen finden Sie unter Verbindung zu Ihrem Amazon Neptune Neptune-Diagramm herstellen.

Um Ihre Daten während der Übertragung zu schützen, erzwingt Neptune mithilfe sicherer Protokolle und Chiffren SSL-Verbindungen über HTTPS zu jeder Instanz oder jedem Cluster-Endpunkt. Neptune stellt SSL-Zertifikate für Ihre Neptune-DB-Instances bereit. Neptune SSL-Zertifikate unterstützen nur Hostnamen für Cluster-Endpunkte, Reader-Endpunkte und Instance-Endpunkte.

Wenn Sie einen Load Balancer oder einen Proxyserver (z. B. HAProxy) verwenden, müssen Sie die SSL-Terminierung verwenden und Ihr eigenes SSL-Zertifikat auf dem Proxyserver haben. SSL-Passthrough funktioniert nicht, da die bereitgestellten SSL-Zertifikate nicht mit dem Hostnamen des Proxy-Servers übereinstimmen. Weitere Informationen zum Herstellen einer Verbindung zu Neptune-Endpunkten mit SSL finden Sie unter Verwenden des HTTP-REST-Endpunkts zur Verbindung mit einer Neptune-DB-Instance.

Implementieren Sie Authentifizierung und Autorisierung

Um zu kontrollieren, wer Neptune-Verwaltungsaktionen auf Neptune-DB-Clustern und DB-Instances ausführen kann, aktivieren Sie die IAM-Datenbankauthentifizierung und verwenden Sie IAM-Anmeldeinformationen. Wenn Sie AWS mithilfe von IAM-Anmeldeinformationen eine Verbindung herstellen, muss Ihre IAM-Rolle über IAM-Richtlinien verfügen, die die für die Ausführung von Neptune-Verwaltungsoperationen erforderlichen Berechtigungen gewähren. Stellen Sie sicher, dass Sie dem Prinzip der geringsten Rechte folgen und nur die Berechtigungen gewähren, die für die Ausführung einer Aufgabe erforderlich sind. Weitere Informationen finden Sie unter

Verschiedene Arten von IAM-Richtlinien zur Steuerung des Zugriffs auf Neptune verwenden und IAM-Authentifizierung mithilfe temporärer Anmeldeinformationen.

Um zu kontrollieren, wer eine Verbindung zu einem Neptune-Cluster herstellen und die Daten abfragen kann, können Sie IAM verwenden, um sich bei Ihrer Neptune-DB-Instance oder Ihrem Neptune-DB-Cluster zu authentifizieren. Wenn Sie die IAM-Authentifizierung in einem Neptune-DB-Cluster aktivieren, muss jeder, der auf den DB-Cluster zugreift, zuerst authentifiziert werden. Weitere Informationen finden Sie unter Aktivieren der IAM-Datenbankauthentifizierung in Neptune für Schritte zur Aktivierung der IAM-Authentifizierung.

Wenn die IAM-Datenbank-Authentifizierung aktiviert ist, muss jede Anforderung mit AWS Signature Version 4 signiert werden. Informationen zum Senden signierter Anfragen an alle Neptune-Endpunkte mit aktivierter IAM-Authentifizierung finden Sie unter Connecting and Signing with Signature Version 4. AWS Viele Bibliotheken und Tools, wie zum Beispiel awscurl, unterstützen bereits Signature Version 4. AWS

Für die Interaktion mit anderen AWS-Services verwendet Amazon Neptune mit dem Service verknüpfte IAM-Rollen. Eine serviceverknüpfte Rolle ist eine spezielle IAM-Rolle, die direkt mit Neptune verknüpft ist. Dienstbezogene Rollen sind von Neptune vordefiniert und beinhalten alle Berechtigungen, die der Dienst benötigt, um andere AWS-Services in Ihrem Namen anzurufen. Weitere Informationen finden Sie unter Verwenden von dienstverknüpften Rollen für Neptune.

Säule der Zuverlässigkeit

Die Säule Zuverlässigkeit umfasst die Fähigkeit eines Workloads, die vorgesehene Funktion korrekt und konsistent auszuführen, wenn dies erwartet wird. Dies umfasst die Möglichkeit, die Workload während des gesamten Lebenszyklus zu betreiben und zu testen.

Ausgangspunkt für eine zuverlässige Workload sind vorab getroffene Designentscheidungen für Software und Infrastruktur. Ihre Architekturentscheidungen wirken sich auf Ihr Workload-Verhalten in allen AWS Well-Architected aus. Zur Gewährleistung von Zuverlässigkeit sind bestimmte Muster zu befolgen.

Die Säule Zuverlässigkeit konzentriert sich auf die folgenden Schlüsselbereiche:

- Workload-Architektur, einschließlich Servicequotas und Bereitstellungsmustern
- Änderungsmanagement
- Fehlerverwaltung

Neptune-Dienstkontingente verstehen

Ein Neptun-Cluster-Volume kann auf eine maximale Größe von 128 Tebibyte (TiB) anwachsen, AWS-Regionen sofern nicht China unterstützt wird und GovCloud wo das Kontingent 64 TiB beträgt.

Das 128 TiB-Kontingent reicht aus, um etwa 200-400 Milliarden Objekte im Diagramm zu speichern. In einem Labeled Property Graph (LPG) ist ein Objekt ein Knoten, eine Kante oder eine Eigenschaft an einem Knoten oder einer Kante. In einem RDF-Diagramm (Resource Description Framework) ist ein Objekt ein Quad.

Für jeden Neptune Serverless-Cluster legen Sie sowohl die minimale als auch die maximale Anzahl von Neptune Capacity Units () fest. NCUs Jede NCU besteht aus 2 Gibibytes (GiB) Arbeitsspeicher und der zugehörigen vCPU und dem Netzwerk. Die minimalen und maximalen NCU-Werte gelten für alle serverlosen Instanzen im Cluster. Der höchste maximale NCU-Wert, den Sie festlegen können, ist 128,0 NCUs, und der niedrigste Mindestwert ist 1,0. NCUs Optimieren Sie den NCU-Bereich, der für Ihre Anwendung am besten geeignet ist, indem Sie die CloudWatch Amazon-Metriken beobachten `ServerlessDatabaseCapacity` und `NCUUtilization` den Bereich erfassen, in dem Sie häufig arbeiten, und unerwünschtes Verhalten oder Kosten innerhalb dieses Bereichs korrelieren. Bei vielen Workloads ist 1,0 NCU ein zu niedriger Startpunkt und führt nach Phasen der Inaktivität zu unzuverlässigem Verhalten. Wenn Sie feststellen, dass Ihr Workload nicht schnell genug skaliert,

erhöhen Sie den Mindestwert, NCUs um ausreichend Rechenleistung für den anfänglichen Anstieg während der Skalierung bereitzustellen.

In AWS-Konto jeder Region gibt es Kontingente für die Anzahl der Datenbankressourcen, die Sie erstellen können. Zu diesen Ressourcen gehören DB-Instances und DB-Cluster. Nachdem eine Größenbeschränkung für eine Ressource erreicht wurde, schlagen zusätzliche Aufrufe zum Erstellen dieser Ressource mit einer Ausnahme fehl. Bei einigen Kontingenten handelt es sich um unverbindliche Kontingente, die auf Anfrage erhöht werden können. Eine Liste der Kontingente, die von Amazon Neptune und Amazon RDS, Amazon Aurora und Amazon DocumentDB gemeinsam genutzt werden (mit MongoDB-Kompatibilität), zusammen mit Links zur Beantragung von Kontingenterhöhungen, sofern verfügbar, finden Sie unter Kontingente in Amazon RDS.

Verstehen Sie die Einsatzmuster von Neptune

In Neptune-DB-Clustern gibt es eine primäre DB-Instance und bis zu 15 Neptune-Repliken. Die primäre DB-Instance unterstützt Lese- und Schreibvorgänge und führt alle Datenänderungen am Cluster-Volume durch. Neptune-Replikate stellen eine Verbindung zu demselben Speichervolume her wie die primäre DB-Instance und unterstützen nur Lesevorgänge. Neptune-Replicas können Lese-Workloads von der primären DB-Instance auslagern.

Verwenden Sie Read Replicas, um eine hohe Verfügbarkeit zu erreichen. Wenn eine oder mehrere Read Replica-Instanzen in verschiedenen Availability Zones verfügbar sind, kann dies die Verfügbarkeit erhöhen, da Read Replicas als Failover-Ziele für die primäre Instanz dienen. Wenn die Writer-Instance ausfällt, befördert Neptune eine Read Replica-Instanz zur primären Instanz. In diesem Fall kommt es zu einer kurzen Unterbrechung (in der Regel weniger als 30 Sekunden), während die hochgestufte Instanz neu gestartet wird. Während dieser Zeit schlagen Lese- und Schreibanforderungen an die primäre Instance mit einer Ausnahme fehl. Für höchste Zuverlässigkeit sollten Sie zwei Read Replicas in verschiedenen Availability Zones in Betracht ziehen. Wenn die primäre Instance in Availability Zone 1 offline geht, wird die Instance in Availability Zone 2 zur primären Instanz heraufgestuft, kann aber keine Abfragen verarbeiten, solange das passiert. Daher ist eine Instance in Availability Zone 3 erforderlich, um Leseabfragen während des Übergangs zu verarbeiten.

Wenn Sie Neptune Serverless verwenden, werden Reader- und Writer-Instances in allen Availability Zones unabhängig voneinander je nach Datenbanklast hoch- und herunterskaliert. Sie können die Promotion-Stufe einer Reader-Instanz auf 0 oder 1 setzen, sodass sie zusammen mit der Kapazität der Writer-Instanz nach oben oder unten skaliert wird. Dadurch ist sie jederzeit bereit, die aktuelle Arbeitslast zu übernehmen.

Wenn Ihre Anwendung weltweit präsent ist oder ein Failover für mehrere Regionen erfordert, sollten Sie die Verwendung einer globalen Neptune-Datenbank in Betracht ziehen. Eine globale Amazon Neptune Neptune-Datenbank erstreckt sich über mehrere AWS-Regionen, ermöglicht globale Lesevorgänge mit geringer Latenz und ermöglicht eine schnelle Wiederherstellung in den seltenen Fällen, in denen ein Ausfall eine gesamte Datenbank betrifft. AWS-Region Eine globale Neptune-Datenbank besteht aus einem primären DB-Cluster in einer Region und bis zu fünf sekundären DB-Clustern in verschiedenen Regionen.

Neptun-Cluster verwalten und skalieren

Sie können Neptune auto-scaling verwenden, um die Anzahl der Neptune-Replikate in einem DB-Cluster automatisch an Ihre Konnektivitäts- und Workload-Anforderungen auf der Grundlage von CPU-Auslastungsschwellenwerten anzupassen. Mit der auto-scaling kann Ihr Neptune-DB-Cluster plötzlichen Anstieg der Arbeitslast bewältigen. Wenn die Arbeitslast abnimmt, entfernt die auto-scaling unnötige Replikate, sodass Sie nicht für ungenutzte Kapazität zahlen müssen. Beachten Sie, dass der Start neuer Instances bis zu 15 Minuten dauern kann. auto-scaling allein ist also keine ausreichende Lösung für schnelle Bedarfsänderungen.

Sie können auto-scaling nur mit einem Neptune-DB-Cluster verwenden, der bereits über eine primäre Writer-Instance und mindestens eine Read-Replica-Instance verfügt (siehe Amazon Neptune DB Clusters and Instances). Außerdem müssen alle Read-Replica-Instances im Cluster verfügbar sein. Wenn sich eine Read-Replica in einem anderen Status als verfügbar befindet, tut Neptune auto-scaling nichts, bis jede Read-Replica im Cluster verfügbar ist.

Wenn sich die Nachfrage schnell ändert, sollten Sie die Verwendung serverloser Instances in Betracht ziehen. Die serverlosen Instances können über kurze Zeiträume vertikal skaliert werden, während die auto-scaling über längere Zeiträume horizontal skaliert wird. Diese Konfiguration bietet optimale Skalierbarkeit, da die serverlosen Instances vertikal skaliert werden, während die auto-scaling neue Read Replicas instanziiert, um die Arbeitslast zu bewältigen, die über die maximale Kapazität einer einzelnen serverlosen Instanz hinausgeht. Weitere Informationen zur Kapazitätsskalierung von Amazon Neptune Serverless finden Sie unter Kapazitätsskalierung in einem Neptune Serverless DB-Cluster.

Wenn sich Ihre Skalierungsanforderungen zu vorhersehbaren Zeiten ändern, können Sie Änderungen der Mindestanzahl an Instances, der maximalen Anzahl von Instances und Schwellenwerten planen, um diesen wechselnden Anforderungen besser gerecht zu werden. Denken Sie daran, Scale-Out-Ereignisse mindestens 15 Minuten im Voraus zu planen, damit diese Instances bei Bedarf online gehen können.

Sie können Ihre Datenbankkonfiguration in Amazon Neptune mittels Parametern in einer Parametergruppe verwalten. Parametergruppen dienen als Container für Engine-Konfigurationswerte, die auf eine oder mehrere DB-Instances angewendet werden. Wenn Sie Cluster-Parameter in Parametergruppen ändern, sollten Sie den Unterschied zwischen statischen und dynamischen Parametern verstehen und wissen, wie und wann sie angewendet werden. Verwenden Sie den Status-Endpunkt, um die aktuell angewendete Konfiguration zu sehen.

Backups und Failover-Ereignisse verwalten

Neptune sichert Ihr Cluster-Volume automatisch und bewahrt die gesicherten Daten für die Dauer des Backup-Aufbewahrungszeitraums auf. Neptune-Sicherungen sind kontinuierlich und inkrementell, so dass Sie schnell eine Sicherung zu einem beliebigen Punkt im Aufbewahrungszeitraum für Sicherungen durchführen können. Sie können einen Aufbewahrungszeitraum für Backups von 1—35 Tagen angeben, wenn Sie einen DB-Cluster erstellen oder ändern.

Um ein Backup über den Aufbewahrungszeitraum des Backups hinaus aufzubewahren, können Sie auch einen Snapshot der Daten in Ihrem Cluster-Volume erstellen. Für das Speichern von Snapshots fallen die Standardgebühren für die Speicherplatznutzung in Neptune an.

Wenn Sie einen Amazon Neptune Neptune-Snapshot eines DB-Clusters erstellen, erstellt Neptune einen Speicher-Volume-Snapshot des Clusters und sichert alle seine Daten, nicht nur einzelne Instances. Sie können einen DB-Cluster erstellen, indem Sie eine Wiederherstellung aus diesem DB-Cluster-Snapshot durchführen. Wenn Sie den DB-Cluster wiederherstellen, geben Sie den Namen des DB-Cluster-Snapshots an, aus dem wiederhergestellt werden soll, und dann geben Sie einen Namen für den neuen DB-Cluster an, der bei der Wiederherstellung erstellt wird.

Testen Sie, wie Ihr System auf Failover-Ereignisse reagiert. Verwenden Sie die Neptune-API, um ein Failover-Ereignis zu erzwingen. Ein Neustart mit Failover ist nützlich, wenn Sie einen Ausfall einer DB-Instance zu Testzwecken simulieren oder um nach einem Failover Operationen in der ursprünglichen Availability Zone wiederherzustellen. Weitere Informationen finden Sie unter Konfiguration und Verwaltung einer Multi-AZ-Bereitstellung. Wenn Sie eine DB-Writer-Instance neu starten, erfolgt ein Failover auf das Standby-Replikat. Das Neustarten eines Neptune-Replikats leitet kein Failover ein.

Gestalten Sie Ihre Clients im Hinblick auf Zuverlässigkeit. Testen Sie ihr Verhalten bei Failover-Ereignissen. Implementieren Sie in Ihrem Client eine Wiederholungslogik mit exponentieller Backoff-Logik. Codebeispiele, die diese Logik implementieren, finden Sie in der Dokumentation unter den AWS Lambda Funktionsbeispielen für Amazon Neptune.

Ziehen Sie die Verwendung in Betracht, AWS Backupwenn Sie gemeinsame Sicherungsanforderungen haben, die Sie für mehrere Datenbank-Engines anwenden.

Säule der Leistungseffizienz

Die Säule der Leistungseffizienz des AWS Well-Architected Framework konzentriert sich darauf, wie die Leistung beim Einlesen oder Abfragen von Daten optimiert werden kann. Die Leistungsoptimierung ist ein inkrementeller und kontinuierlicher Prozess, der Folgendes umfasst:

- Bestätigung der Geschäftsanforderungen
- Messung der Workload-Leistung
- Identifizierung leistungsschwacher Komponenten
- Abstimmung der Komponenten auf Ihre Geschäftsanforderungen

Im Bereich Leistungseffizienz finden Sie anwendungsfallspezifische Richtlinien, die Ihnen dabei helfen können, das richtige Grafikdatenmodell und die richtigen Abfragesprachen zu finden. Es enthält auch bewährte Methoden, die bei der Aufnahme von Daten in Amazon Neptune und der Nutzung von Daten aus Amazon Neptune zu beachten sind.

Der Schwerpunkt der Leistungseffizienz konzentriert sich auf die folgenden Schlüsselbereiche:

- Graphmodellierung
- Optimierung von Abfragen
- Richtige Clustergröße
- Optimierung schreiben

Verstehen Sie die Graphmodellierung

Verstehen Sie den Unterschied zwischen den Modellen Labeled Property Graph (LPG) und Resource Description Framework (RDF). In den meisten Fällen ist dies eine Frage der Präferenz. Es gibt jedoch mehrere Anwendungsfälle, in denen ein Modell besser geeignet ist als das andere. Wenn Sie den Pfad kennen müssen, der zwei Knoten in Ihrem Diagramm verbindet, wählen Sie LPG. Wenn Sie Daten aus Neptun-Clustern oder anderen Graph Triple Stores zusammenführen möchten, wählen Sie RDF.

Wenn Sie eine SaaS-Anwendung (Software as a Service) oder eine Anwendung entwickeln, die Mehrmandantenfähigkeit erfordert, sollten Sie die logische Trennung von Mandanten in Ihr Datenmodell integrieren, anstatt einen Mandanten für jeden Cluster zu haben. Um diese Art von Design zu erreichen, können Sie benannte SPARQL-Diagramme und Kennzeichnungsstrategien

verwenden, z. B. Kundenkennungen den Bezeichnungen voranstellen oder Eigenschaftsschlüssel-Wert-Paare hinzufügen, die Mandantenkennungen darstellen. Stellen Sie sicher, dass Ihre Client-Ebene diese Werte einfügt, um diese logische Trennung beizubehalten. Weitere Informationen zu Empfehlungen für mehrere Mandanten finden Sie unter Multi-Tenancy-Richtlinien für den ISVs Betrieb von Amazon Neptune Neptune-Datenbanken.

Die Leistung Ihrer Abfragen hängt von der Anzahl der Diagrammobjekte (Knoten, Kanten, Eigenschaften) ab, die bei der Verarbeitung Ihrer Abfrage ausgewertet werden müssen. Daher kann das Graphmodell erhebliche Auswirkungen auf die Leistung Ihrer Anwendung haben. Verwenden Sie nach Möglichkeit detaillierte Beschriftungen und speichern Sie nur die Eigenschaften, die Sie für die Pfadbestimmung oder Filterung benötigen. Um eine höhere Leistung zu erzielen, sollten Sie erwägen, Teile Ihres Diagramms vorab zu berechnen, z. B. Zusammenfassungsknoten oder direktere Kanten zu erstellen, die gemeinsame Pfade verbinden.

Vermeiden Sie es, zwischen Knoten zu navigieren, die eine ungewöhnlich hohe Anzahl von Kanten mit derselben Bezeichnung haben. Solche Knoten haben oft Tausende von Kanten (wobei die meisten Knoten eine Kantenanzahl im Zehnerbereich haben). Das Ergebnis ist eine viel höhere Rechen- und Datenkomplexität. Diese Knoten sind bei einigen Abfragemustern möglicherweise nicht problematisch, wir empfehlen jedoch, Ihre Daten anders zu modellieren, um dies zu vermeiden, insbesondere wenn Sie als Zwischenschritt über den Knoten navigieren. Sie können Protokolle für langsame Abfragen verwenden, um Abfragen zu identifizieren, die zwischen diesen Knoten navigieren. Sie werden wahrscheinlich deutlich höhere Latenz- und Datenzugriffsmetriken als Ihre durchschnittlichen Abfragemuster beobachten, insbesondere wenn Sie den Debug-Modus verwenden.

Verwenden Sie deterministischen Knoten IDs für Knoten und Kanten, wenn Ihr Anwendungsfall dies unterstützt, anstatt Neptune zu verwenden, um zufällige GUID-Werte zuzuweisen. IDs Der Zugriff auf Knoten anhand der ID ist die effizienteste Methode.

Abfragen optimieren

Die Sprachen OpenCypher und Gremlin können auf LPG-Modellen synonym verwendet werden. Wenn Leistung ein Hauptanliegen ist, sollten Sie erwägen, die beiden Sprachen synonym zu verwenden, da eine Sprache bei bestimmten Abfragemustern möglicherweise besser abschneidet als die andere.

Neptune ist dabei, zu seiner Alternative Query Engine (DFE) zu konvertieren. OpenCypher läuft nur auf dem DFE, aber sowohl Gremlin- als auch SPARQL-Abfragen können optional so eingestellt

werden, dass sie auf dem DFE ausgeführt werden, indem Abfrageanmerkungen verwendet werden. Erwägen Sie, Ihre Abfragen mit aktiviertem DFE zu testen und die Leistung Ihres Abfragemusters zu vergleichen, wenn Sie DFE nicht verwenden.

Neptune ist für transaktionale Abfragen optimiert, die an einem einzelnen Knoten oder einer Gruppe von Knoten beginnen und sich von dort aus ausbreiten, und nicht für analytische Abfragen, die den gesamten Graphen auswerten. Verwenden Sie Neptune Analytics für Ihre analytischen Abfrage-Workloads. Neptune Analytics ist die ideale Wahl für investigative, explorative oder datenwissenschaftliche Workloads, die eine schnelle Iteration für die Daten-, analytische und algorithmische Verarbeitung erfordern. Es kann auch eine Vektorsuche in Grafikdaten durchführen und Daten direkt aus Ihrer Neptune-Datenbankinstanz laden. Wenn Neptune Analytics Ihren Anforderungen nicht entspricht, können Sie auch ein AWS SDK für Pandas oder die Verwendung von Neptune-Export in Kombination mit Amazon EMR in Betracht ziehen. AWS Glue

Um Ineffizienzen und Engpässe in Ihren Modellen und Abfragen zu identifizieren, verwenden Sie die Option und `explain` APIs für jede Abfragesprache, um detaillierte Erläuterungen zum Abfrageplan `profile` und zu den Abfragemetriken zu erhalten. Weitere Informationen finden Sie unter Gremlin-Profil, OpenCypher Explain und SPARQL Explain.

Verstehen Sie Ihre Abfragemuster. Wenn die Anzahl der unterschiedlichen Kanten in einem Diagramm groß wird, kann die standardmäßige Neptun-Zugriffsstrategie ineffizient werden. Die folgenden Abfragen könnten ziemlich ineffizient werden:

- Abfragen, die rückwärts über Kanten navigieren, wenn keine Kantenbeschriftungen angegeben sind.
- Klauseln, die intern dasselbe Muster verwenden, z. B. `.both()` in Gremlin, oder Klauseln, die Knoten in einer beliebigen Sprache löschen (was das Löschen eingehender Kanten ohne Kenntnis der Labels erfordert).
- Abfragen, die auf Eigenschaftswerte zugreifen, ohne Eigenschaftsbezeichnungen anzugeben. Diese Abfragen könnten ziemlich ineffizient werden. Wenn dies Ihrem Nutzungsmuster entspricht, sollten Sie erwägen, den OSGP-Index (Objekt, Subjekt, Diagramm, Prädikat) zu aktivieren.

Verwenden Sie die Protokollierung langsamer Abfragen, um langsame Abfragen zu identifizieren. Langsame Abfragen können durch nicht optimierte Abfragepläne oder unnötig viele Indexsuchen verursacht werden, was die Kosten in die Höhe treiben kann. I/O Die Neptune Explain and Profile Endpoints für Gremlin, SPARQL oder OpenCypher können Ihnen helfen zu verstehen, warum diese Abfragen langsam sind. Zu den möglichen Ursachen gehören:

- Knoten mit einer ungewöhnlich hohen Anzahl von Kanten im Vergleich zum durchschnittlichen Knoten im Diagramm (z. B. Tausende im Vergleich zu Zehnern) können die Rechenkomplexität erhöhen und somit die Latenz und den Ressourcenverbrauch erhöhen. Stellen Sie fest, ob diese Knoten korrekt modelliert sind oder ob die Zugriffsmuster verbessert werden können, um die Anzahl der Kanten, die überquert werden müssen, zu reduzieren.
- Nicht optimierte Abfragen enthalten eine Warnung, dass bestimmte Schritte nicht optimiert sind. Das Umschreiben dieser Abfragen zur Verwendung optimierter Schritte kann die Leistung verbessern.
- Redundante Filter können zu unnötigen Indexsuchen führen. Ebenso können redundante Muster zu doppelten Indexsuchen führen, die durch eine Verbesserung der Abfrage optimiert werden können (siehe `Index Operations - Duplication ratio` in der Profilausgabe).
- In einigen Sprachen wie Gremlin gibt es keine stark typisierten numerischen Werte und sie verwenden stattdessen die Typ-Heraufstufung. Wenn der Wert beispielsweise 55 ist, sucht Neptune nach Werten, die Ganzzahlen, Longs, Gleitkommazahlen und andere numerische Typen sind, die 55 entsprechen. Dies führt zu zusätzlichen Operationen. Wenn Sie im Voraus wissen, dass Ihre Typen übereinstimmen, können Sie dies vermeiden, indem Sie einen Abfragehinweis verwenden.
- Ihr Grafikmodell kann sich erheblich auf die Leistung auswirken. Erwägen Sie, die Anzahl der Objekte, die ausgewertet werden müssen, zu reduzieren, indem Sie detailliertere Beschriftungen verwenden oder Abkürzungen für lineare Multiple-Hop-Pfade im Voraus berechnen.

Wenn die Abfrageoptimierung allein es Ihnen nicht ermöglicht, Ihre Leistungsanforderungen zu erfüllen, sollten Sie erwägen, verschiedene Caching-Techniken mit Neptune zu verwenden, um diese Anforderungen zu erfüllen.

Die Leistung von Neptune verbessert sich mit jeder Version kontinuierlich. In den Versionshinweisen finden Sie Einzelheiten zu den Verbesserungen mit jeder Version. Erwägen Sie, regelmäßige Updates für Ihren Neptune DB-Cluster zu planen, um eine optimale Leistung zu erzielen. Neuere Versionen unterstützen auch neuere Instances. Erwägen Sie ein Upgrade auf 1.4.5.0 oder höher, um die `r8g` Instanzen nutzen zu können. Weitere Informationen darüber, wie dies die Leistung Ihres Workloads verbessern kann, finden Sie unter 4,7-mal besseres Preis-Leistungs-Verhältnis für Schreibabfragen mit AWS Graviton4 R8g-Instances mit Amazon Neptune v1.4.5.

Cluster mit der richtigen Größe

Passen Sie die Größe Ihres Clusters an Ihre Anforderungen an Parallelität und Durchsatz an. Die Anzahl der gleichzeitigen Abfragen, die von jeder Instance im Cluster verarbeitet werden können, entspricht dem Zweifachen der Anzahl virtueller Abfragen CPUs (vCPUs) auf dieser Instance. Zusätzliche Abfragen, die eintreffen, während alle Worker-Threads belegt sind, werden in eine serverseitige Warteschlange gestellt. Diese Abfragen werden auf FIFO-Basis bearbeitet, wenn Worker-Threads verfügbar werden. first-in-first-out Die `MainRequestQueuePendingRequests` CloudWatch Amazon-Metrik zeigt die aktuelle Warteschlangentiefe für jede Instance.
Wenn dieser Wert häufig über Null liegt, sollten Sie eine Instance mit mehr v wählenCPUs.
Wenn die Warteschlangentiefe 8.192 überschreitet, gibt Neptune einen Fehler zurück. `ThrottlingException`

Ungefähr 65 Prozent des RAM für jede Instanz sind für den Puffercache reserviert. Der Puffercache enthält den Arbeitsdatensatz (nicht das gesamte Diagramm, sondern nur die Daten, die abgefragt werden). Überwachen Sie die Metrik, um festzustellen, welcher Prozentsatz der Daten aus dem Puffer-Cache und nicht aus dem Speicher abgerufen wird. CloudWatch `BufferCacheHitRatio`
Wenn diese Metrik häufig unter 99,9 Prozent fällt, sollten Sie es mit einer Instance mit mehr Arbeitsspeicher versuchen, um festzustellen, ob dadurch Ihre Latenz und I/O Ihre Kosten gesenkt werden.

Read Replicas müssen nicht dieselbe Größe wie Ihre Writer-Instance haben. Starke Schreiblasten können jedoch dazu führen, dass kleinere Replikate ins Hintertreffen geraten und neu gestartet werden, weil sie mit der Replikation nicht Schritt halten können. Aus diesem Grund empfehlen wir, Replikate gleich oder größer als die Writer-Instance zu erstellen.

Wenn Sie auto-scaling für Ihre Read Replicas verwenden, denken Sie daran, dass es bis zu 15 Minuten dauern kann, bis eine neue Read Replica online ist. Wenn der Client-Verkehr schnell, aber vorhersehbar zunimmt, sollten Sie eine geplante Skalierung in Betracht ziehen, um die Mindestanzahl von Read Replicas entsprechend dieser Initialisierungszeit zu erhöhen.

Serverlose Instanzen unterstützen verschiedene Anwendungsfälle und Workloads. Ziehen Sie in den folgenden Szenarien serverlose statt bereitgestellte Instanzen in Betracht:

- Ihre Arbeitslast schwankt im Laufe des Tages häufig.
- Sie haben eine neue Anwendung erstellt und sind sich nicht sicher, wie groß die Arbeitslast sein wird.
- Sie entwickeln und testen.

Es ist wichtig zu beachten, dass serverlose Instanzen teurer sind als vergleichbare bereitgestellte Instanzen, wenn man einen Dollar pro GB RAM berechnet. Jede serverlose Instanz besteht aus 2 GB RAM zusammen mit der zugehörigen vCPU und dem Netzwerk. Führen Sie eine Kostenanalyse zwischen Ihren Optionen durch, um überraschende Rechnungen zu vermeiden. Im Allgemeinen erzielen Sie mit Serverless nur dann Kosteneinsparungen, wenn Ihre Arbeitslast nur wenige Stunden am Tag sehr hoch ist und den Rest des Tages fast Null ist oder wenn Ihre Arbeitslast im Laufe des Tages stark schwankt.

Verwenden Sie den Amazon Neptune Neptune-Preisrechner, um anhand von Faktoren wie queries-per-second (QPS) -Anforderungen die richtige Konfiguration für Ihren Cluster zu ermitteln.

Schreibvorgänge optimieren

Beachten Sie Folgendes, um Schreibvorgänge zu optimieren:

- Der Neptune Bulk Loader ist die optimale Methode, um Ihre Datenbank zunächst zu laden oder an bestehende Daten anzuhängen. Der Neptune-Loader ist nicht transaktional und kann keine Daten löschen. Verwenden Sie ihn daher nicht, wenn dies Ihre Anforderungen sind.
- Transaktionsaktualisierungen können mithilfe der unterstützten Abfragesprachen vorgenommen werden. Um I/O Schreibvorgänge zu optimieren, schreiben Sie Daten in Stapeln von 50 bis 100 Objekten pro Commit. Ein Objekt ist ein Knoten, eine Kante oder eine Eigenschaft an einem Knoten oder einer Kante in LPG oder ein Triple Store oder ein Quad in RDF.
- Alle transaktionalen Neptune-Schreiboperationen werden für jede Verbindung in einem Thread ausgeführt. Wenn Sie eine große Datenmenge an Neptune senden, sollten Sie mehrere parallel Verbindungen in Betracht ziehen, die jeweils Daten schreiben. Wenn Sie sich für eine von Neptune bereitgestellte Instanz entscheiden, wird die Instanzgröße einer Zahl von v zugeordnet. CPUs Neptune erstellt zwei Datenbank-Threads für jede vCPU auf der Instance. Beginnen Sie also mit der doppelten Anzahl von v, CPUs wenn Sie die optimale Parallelisierung testen. Serverlose Instanzen skalieren die Anzahl von v mit einer CPUs Rate von ungefähr eins für jede 4. NCUs

  > **Note**
  >
  > Dies gilt nicht für die Massen-Loading-API, sondern nur für Direktverbindungen.

- Planen Sie alle Schreibvorgänge ein ConcurrentModificationExceptionsund wickeln Sie sie effizient ab, auch wenn zu einem beliebigen Zeitpunkt nur eine einzige Verbindung

Daten schreibt. Gestalten Sie Ihre Clients so, dass sie zuverlässig sind, wenn sie `ConcurrentModificationExceptions` auftreten.

- Wenn Sie alle Ihre Daten löschen möchten, sollten Sie die Fast-Reset-API verwenden, anstatt gleichzeitig Löschabfragen zu stellen. Letzteres wird im Vergleich zu Ersterem viel länger dauern und erhebliche I/O Kosten verursachen.
- Wenn Sie die meisten Ihrer Daten löschen möchten, sollten Sie erwägen, die Daten, die Sie behalten möchten, zu exportieren, indem Sie die Daten mithilfe von neptune-export in einen neuen Cluster laden. Löschen Sie dann den ursprünglichen Cluster.

Säule der Kostenoptimierung

Die Säule der Kostenoptimierung des AWS Well-Architected Framework konzentriert sich auf die Vermeidung unnötiger Kosten. Die folgenden Empfehlungen können Ihnen helfen, die Entwurfsprinzipien zur Kostenoptimierung und die bewährten Architekturpraktiken für Amazon Neptune zu erfüllen.

Die Säule Kostenoptimierung konzentriert sich auf die folgenden Schlüsselbereiche:

- Überblick über die Ausgaben im Zeitverlauf und Kontrolle der Mittelzuweisung
- Auswahl von Ressourcen der richtigen Art und Menge
- Skalierung zur Erfüllung der Geschäftsanforderungen ohne Mehrausgaben

Verstehen Sie die Nutzungsmuster und die benötigten Dienste

Neptune eignet sich gut für Ihren Workload, wenn Ihr Datenmodell eine erkennbare Graphstruktur hat und Ihre Abfragen Beziehungen untersuchen und mehrere Hops durchlaufen müssen. Eine Graphdatenbank eignet sich nicht für die folgenden Muster:

- Hauptsächlich Single-Hop-Abfragen (überlegen Sie, ob Ihre Daten besser als Attribute eines Objekts dargestellt werden könnten)
- JSON- oder BLOB-Daten, die als Eigenschaften gespeichert werden
- Abfragen, die sich über einen Datensatz hinweg aggregieren, z. B. die Berechnung der Summe einer numerischen Eigenschaft über eine große Anzahl von Knoten

Überlegen Sie, ob die gemeinsame Verwendung mehrerer speziell entwickelter Datenbanken für bestimmte Zugriffsmuster all Ihre Anforderungen erfüllen könnte. Beispiel:

- Eine API, die weniger häufige komplexe Graphnavigationen erfordert und gleichzeitig Eigenschaften für einen einzelnen Knoten gleichzeitig abgerufen werden muss, lässt sich am besten mit einem oder mehreren von Neptune, DynamoDB oder Amazon DocumentDB präsentieren.
- Relationale Datenbanken können mit Neptune koexistieren, um Ihre bestehende Funktionalität beizubehalten. Verwenden Sie Neptune jedoch nur für Multiple-Hop-Traversals, die in relationalen Datenbanken nicht gut funktionieren und nicht gut skalieren.

Machen Sie sich mit den Kosten vertraut, die mit Diensten verbunden sind, die mit Neptune interagieren und diese ergänzen, einschließlich der folgenden:

- Speicherkosten für Amazon Simple Storage Service (Amazon S3) für Datendateien, die massenweise in Neptune geladen werden
- Lambda-Funktionen, die für Insert- oder Upsert-Abfragen, Leseabfragen und die Verarbeitung von Neptune-Streams verwendet werden
- Die auf Neptune aufgebaute API-Schicht zur Interaktion mit der Client-Anwendung (anstatt direkte Verbindungen zur Datenbank herzustellen) in Amazon API Gateway oder AWS AppSync
- AWS Glue Jobs, die zum Übertragen von Daten zu und von Neptune verwendet werden
- Amazon Kinesis- oder Amazon Managed Streaming for Apache Kafka (Amazon MSK) -Instances empfangen Streaming-Daten für die Aufnahme nahezu in Echtzeit in Neptune.
- AWS Database Migration Service für die Migration relationaler Daten nach Neptune
- Amazon SageMaker Runtime-Kosten für Jupyter-Notebooks und Machine-Learning-Modelle mit Deep Graph Library

Wählen Sie Ressourcen unter Berücksichtigung der Kosten aus

Die Neptune-Preise basieren auf den stündlichen Instanzkosten (oder den Verbrauch von Neptune-Recheneinheiten bei serverlosem Betrieb), Daten-I/O und Speichernutzung. Instanzen machen im Durchschnitt 85 Prozent der Gesamtkosten aus, sodass die richtige Dimensionierung erhebliche Auswirkungen auf die Kosten haben kann. Die beste Methode zur richtigen Größe von Instances besteht darin, die Anwendungsleistung auf einer Vielzahl von Instances zu testen und die folgenden Faktoren zu vergleichen:

- Bleibt die `MainRequestQueuePendingRequests` CloudWatch Metrik auf einem konstant niedrigen Wert nahe Null?
- Bleibt die `BufferCacheHitRatio` CloudWatch Kennzahl die meiste Zeit bei oder über 99,9 Prozent?
- Wie sehen die Kosten- und Leistungskurven für Instanzkosten und die damit verbundenen I/O Datenkosten aus? Die Kosten für das Lesen von Daten können bei einer unterdimensionierten Instance, die ein häufiges Austauschen des Puffer-Caches mit dem Speicher erfordert, erheblich steigen. `BufferCacheHitRatio`werden in diesen Szenarien häufig sinken.

Die Instanzkosten innerhalb derselben Instance-Familie skalieren linear mit der Größe. Die Stundenkosten der `db.r6i.2xlarge` Instance sind doppelt so hoch wie die der `db.r6i.xlarge` Instance und haben zudem die doppelte Ressourcenzuweisung. Die `db.r6i.24xlarge` Instanz kostet das 24-fache der stündlichen Kosten der `db.r6i.xlarge` Instanz.

Schätzen Sie die Anzahl der gleichzeitigen Abfragen, die Sie unterstützen müssen. Sie können zwischen null und fünfzehn Read Replicas für die Verarbeitung schreibgeschützter Abfragen verwenden. Wenn Ihre Anforderungen je nach Tages-, Wochen- oder Monatszeit variieren, können Sie mehrere kleinere Instances verwenden, um nach einem Zeitplan zu skalieren. Jede vCPU auf einer Instanz stellt zwei Threads für die Verarbeitung gleichzeitiger Abfragen bereit. Drei `db.r6i.xlarge` Read Replicas mit jeweils 4 vCPUs können 24 gleichzeitige Abfragen verarbeiten.

Wenn Ihr Datenverkehrsvolumen stattdessen in Abfragen pro Sekunde (QPS) gemessen wird, müssen Sie experimentieren, um die durchschnittliche Latenz Ihrer Abfragen zu ermitteln. Die Anzahl der Abfragen pro Sekunde, die ein Neptun-Cluster unterstützen kann, entspricht. `vCPU × 2 × (1 second/average query latency)` Wenn Sie beispielsweise über 4 vCPUs und eine Abfragelatenz von 100 Millisekunden (0,1 Sekunden) verfügen,. `QPS = 4 × 2 × (1s/0.1s) = 80 queries per second`

Für kontinuierliche, stabile und vorhersehbare Workloads sind bereitgestellte Instanzen günstiger als serverlose Instanzen. Serverless bietet Möglichkeiten zur Kostenoptimierung, wenn Sie eine Arbeitslast haben, die nur wenige Stunden pro Tag sehr stark ausgelastet ist (z. B.`db.r6i.4xlarge`) und dann für den Rest des Tages fast keinen Verkehr erfordert (z. B. 1 Neptune Compute Unit). Eine serverlose Instanz, die für einige Stunden hochskaliert und dann wieder heruntergefahren wird, ist günstiger als die ganztägige Nutzung einer bereitgestellten `db.r6i.4xlarge` Instanz.

Erwägen Sie ein Upgrade auf Neptune 1.4.5.0 oder höher und die Nutzung von `r8g` Instances, um einen besseren Lese- und Schreibdurchsatz zu geringeren Kosten als Instances älterer Generationen wie oder zu erzielen. `r7g` `r6g` Weitere Informationen finden Sie unter 4,7-mal besseres Preis-Leistungs-Verhältnis beim Schreiben von Abfragen mit AWS Graviton4 R8g-Instances mit Amazon Neptune v1.4.5 (Blogbeitrag).AWS

Neptune-Cluster werden standardmäßig mit Standardspeicher erstellt (wenn Sie sie mit der Konsole erstellen, wählt sie standardmäßig I/O-optimized storage). With I/O-optimized storage, you pay a slightly higher cost for storage and instances, but there are no I/O costs. This leads to more predictable recurring costs, but if your I/O usage is generally low, it may be more cost efficient to utilize standard storage. If you intend to load a lot of data initially, you can optimize cost by choosing I/O -optimierten Speicher aus, führt den ersten Datenladevorgang durch und wechselt dann zum

Standardspeicher. Der Speichertyp wirkt sich nur auf das Abrechnungsmodell aus und hat keinen technischen Unterschied in der Neptune-DB-Cluster- oder Instance-Konfiguration. Sie können den Speichertyp einmal alle 30 Tage ändern. Überprüfe nach 30 Tagen deine detaillierten Neptun-Kosten und berechne anhand der Neptune-Preisseite, ob deine Kosten mit -optimized höher gewesen wären. I/O-optimized storage. If they would have been, continue to use standard storage, otherwise switch back to I/O

Wählen Sie die beste Neptune-Instanzkonfiguration für Ihren Workload

Wenn du dein Spiel AWS-Konto vor dem 15. Juli 2025 erstellt hast, kannst du das AWS kostenlose Kontingent für Experimente mit Neptune auf Einstiegsebene nutzen. Die 750 kostenlosen Stunden `db.t3.medium` und die Nutzung der `db.t4g.medium` Instanzen reichen aus, um Neptune in geringem Umfang gut zu verstehen. Ihr Cluster bleibt auch nach Ablauf der kostenlosen Testphase bestehen, obwohl Ihnen ab diesem Zeitpunkt die Nutzung in Rechnung gestellt wird.

Die `db.t4g.medium` Instanzen `db.t3.medium` und eignen sich gut für kostengünstige Entwicklungsumgebungen, in denen Sie OpenCypher, Graph Explorer oder verschiedene generative KI-Integrationen nicht verwenden. Diese Instanzen haben ein kleineres RAM-to-vCPU Verhältnis (2:1) als die `R` Familieninstanzen (8:1) oder `X` Familieninstanzen (16:1). Dies reduziert das Verhältnis und verhindert die Verwendung von Statistiken der DFE-Engine, die die Leistung von OpenCypher, GenAI-Integrationen (um das LLM über das Graphschema zu informieren) und Graph Explorer. Bei der Verwendung von `T` Familieninstanzen können sich die Leistungsprofile erheblich unterscheiden, insbesondere bei den zuvor genannten Workloads. Diese Instanzen können auch dazu führen, dass Abfragen häufiger vorkommen`OutOfMemoryExceptions`, wenn sie sich über einen wesentlichen Teil des Diagramms bewegen. Überprüfen Sie die `BufferCacheHitRatio` CloudWatch Metrik, um festzustellen, ob die letztgenannte Bedingung betroffen sein könnte.

Wir raten dringend davon ab, Leistungs- oder Lasttests mit `T` Familieninstanzen durchzuführen, da es zu inkonsistenten Ergebnissen kommen kann, die nicht auf eine Produktionsumgebung hinweisen.

Bereitgestellte Instanzen bieten Ihnen die beste Kombination aus Kosten und Leistung, wenn Ihre Arbeitslast relativ stabil und vorhersehbar ist. Wählen Sie die Instanzgröße auf der Grundlage der erforderlichen Parallelität der Anfragen und der Komplexität der Abfrage. Für eine höhere Parallelität ist mehr v erforderlich. CPUs Eine höhere Abfragekomplexität erfordert mehr RAM. Ermitteln Sie anhand der `MainRequestQueuePendingRequests` CloudWatch Metrik, wie sich Ersteres auswirkt (ein Wert größer als Null steht für mehr gleichzeitige Anfragen, als bearbeitet werden können).

Verwenden Sie die `BufferCacheHitRatio` CloudWatch Metrik, um die Auswirkung der letzteren zu ermitteln. Ein Verhältnis, das häufig unter 99,9 Prozent fällt, deutet darauf hin, dass nicht genug RAM vorhanden ist, um den aktiven Teil des auszuwertenden Diagramms aufzunehmen, was zu häufigerem Cache-Swapping führt. Wenn die R-Instance-Familie ausreichend Parallelität, aber nicht genug RAM bietet, sollten Sie die `X` Instance-Familie ausprobieren.

Ideale Anwendungsfälle für serverlose Instanzen sind in der Neptune-Dokumentation beschrieben. Wenn Sie sich nicht sicher sind, ob bereitgestellte oder serverlose Workloads für Sie am besten geeignet sind und die Kosten Ihr Hauptanliegen sind, testen Sie Ihren Workload im serverlosen Modus, um die Anzahl der NCUs genutzten Workloads zu ermitteln und die Kosten für bereitgestellte () mit denen von serverlos () zu vergleichen. `N hours × hourly provisioned cost sum of NCUs × hourly cost per NCU` Wenn Sie sich nicht sicher sind, welche Provision-Instanz die entsprechende Größe hat, entspricht eine NCU etwa 2 GB RAM und der zugehörigen vCPU und dem Netzwerk. Wenn Ihre bereitgestellte Instanz aus der `r6i` Familie stammt, beträgt das Verhältnis 1 vCPU pro 8 GB RAM oder 4 NCUs, zusammen mit dem zugehörigen Netzwerk. Der Amazon Neptune Neptune-Preisrechner bietet auch einen Vergleich, der Ihnen bei der Entscheidung für Ihre optimale Kostenkonfiguration hilft.

Wenn Sie Serverless für Primär- und Replikat-Instances verwenden, denken Sie daran, dass Read Replicas der Promotion-Stufen 0 und 1 entsprechend der Writer-Instance skaliert werden, sodass sie bei einem Failover-Ereignis ordnungsgemäß skaliert werden. NCUs Legen Sie Ihre NCU-Grenzwerte für diese Instances danach fest, welche Ihrer Instances — Writer oder Reader — den meisten Traffic erhalten.

In Umgebungen, in denen der Cluster nicht 24 Stunden am Tag, 7 Tage die Woche benötigt wird, sollten Sie in Erwägung ziehen, Skripte zu schreiben, die die Neptune-Instanzen ausschalten, wenn sie nicht verwendet werden, und sie erneut starten, bevor sie verwendet werden. Neptune-Instanzen werden automatisch alle 7 Tage neu gestartet, um sicherzustellen, dass die erforderlichen Wartungsupdates installiert werden. Wenn Sie beabsichtigen, die Instanzen für längere Zeit ausgeschaltet zu lassen, verwenden Sie ein wöchentliches Skript, um sie wieder herunterzufahren.

Datenspeicherung und -übertragung in der richtigen Größe

Effizientere Abfragen (z. B. Abfragen, die weniger Knoten, Kanten und Eigenschaften im Diagramm berühren müssen) erfordern weniger I/O Übertragung und können möglicherweise kleinere Instanzen verwenden, da weniger Puffer-Cache erforderlich ist. Verwenden Sie das Profil oder die Explain-Endpunkte für Ihre Abfragesprache, um Ihre Abfrage zu optimieren, und ziehen Sie in Betracht, Ihr Grafikmodell im Hinblick auf Ihre Abfrageleistung zu optimieren.

Neptune verwendet die Wörterbuchkodierung für große Zeichenketten, und dieses Wörterbuch ist für Leistung und nicht für Effizienz optimiert. Wenn Sie große BLOBs, JSON-Zeichenketten oder häufig wechselnde Zeichenketten für Eigenschaften haben, sollten Sie erwägen, diese außerhalb von Neptune in Amazon S3, Amazon DynamoDB oder Amazon DocumentDB zu speichern und nur eine Referenz innerhalb des Neptune-Knotens zu speichern.

In einigen Fällen kann es günstiger sein, eine größere Instance-Größe zu wählen. Wenn Ihre I/O Kosten aufgrund einer niedrigen Version sehr hoch sind`BufferCacheHitRatio`, ist es möglich, dass der größere Puffer-Cache diese Kosten erheblich reduzieren würde. Das liegt daran, dass alle Daten in den Cache passen würden, anstatt häufig aus dem Speicher ausgelagert zu werden und die I/O Übertragungsrate zu erhöhen.

Neptune verwendet copy-on-write Klonen. Beim Klonen, um ein Diagramm in mehrere Shards aufzuteilen, ist es möglicherweise effizienter, die unerwünschten Daten auf dem geklonten Cluster nicht zu löschen, da dafür neue Datenseiten erstellt werden müssen, was zu erhöhten Speicherkosten führt. Daten, die gegenüber der Zeit vor dem Klonen unverändert geblieben sind, befinden sich auf einer einzigen Datenseite, die von den beiden Clustern gemeinsam genutzt wird, und es fallen nur Gebühren für diese einzelne Kopie an.

Aktivieren Sie den OSGP-Index nicht und verwenden Sie keine R5d-Instances, es sei denn, Sie haben getestet, um zu bestätigen, dass sie einen wesentlichen Unterschied in Ihrer Arbeitslast bewirken. Beide sind für selten auftretende Szenarien konzipiert und können Ihre Kosten erhöhen, wenn Sie nur minimale oder gar keine Gewinne erzielen.

Säule der Nachhaltigkeit

Der Schwerpunkt der Nachhaltigkeit liegt auf der Minimierung der Umweltauswirkungen der Ausführung von Cloud-Workloads. Zu den wichtigsten Themen gehören ein Modell der gemeinsamen Verantwortung für Nachhaltigkeit, das Verständnis der Auswirkungen und die Maximierung der Nutzung, um die benötigten Ressourcen zu minimieren und die nachgelagerten Auswirkungen zu reduzieren.

Die Säule Nachhaltigkeit umfasst die folgenden Schwerpunktbereiche:

- Ihr Einfluss
- Nachhaltigkeitsziele
- Maximierte Nutzung
- Antizipierung und Einführung neuer, effizienterer Hardware- und Softwareangebote
- Nutzung von Managed Services
- Reduzierung der nachgelagerten Auswirkungen

Dieser Leitfaden konzentriert sich auf Ihre Wirkung. Weitere Informationen zu den anderen Prinzipien des Nachhaltigkeitsdesigns finden Sie im AWS Well-Architected Framework.

Ihre Entscheidungen und Anforderungen wirken sich auf die Umwelt aus. Wenn Sie sich für eine geringere Kohlenstoffintensität entscheiden AWS-Regionen können und Ihre Anforderungen den tatsächlichen Arbeitsanforderungen entsprechen, anstatt nur die Verfügbarkeit und Haltbarkeit zu maximieren, erhöht sich die Nachhaltigkeit der Arbeitslast. In den nächsten Abschnitten werden bewährte Verfahren und durchdachte Überlegungen erörtert, die sich positiv auf die Umwelt auswirken können, wenn sie bei der Planung Ihrer Arbeitslast und Ihrem laufenden Betrieb berücksichtigt werden.

AWS-Region Auswahl

Einige AWS-Regionen befinden sich in der Nähe von Amazonas-Projekten für erneuerbare Energien oder dort, wo das Netz eine veröffentlichte Kohlenstoffintensität aufweist, die niedriger ist als bei anderen. Berücksichtigen Sie die Auswirkungen auf die Nachhaltigkeit der Regionen, die für Ihre Arbeitsbelastung in Frage kommen könnten, und vergleichen Sie Ihre Liste mit den Regionen, in denen Neptune verfügbar ist.

Der Konsum basiert auf Verhaltensmustern der Nutzer

Wenn Sie Ihren Verbrauch an den Traffic und das Verhalten Ihrer Nutzer anpassen, können Sie die Auswirkungen von Diensten auf die Umwelt AWS minimieren. Beachten Sie bei der Entwicklung Ihrer Lösung die folgenden bewährten Methoden:

- Überwachen Sie CloudWatch Amazon-Metriken wie `CPUUtilization`, `MainRequestQueuePendingRequests`, und, `TotalRequestsPerSec` um festzustellen, wann Ihr Bedarf am höchsten und am niedrigsten ist, und stellen Sie sicher, dass Ihre Cluster-Ressourcen in diesen Zeiten die richtige Größe haben.
- Automatisieren Sie das Stoppen von Umgebungen außerhalb der Produktion während der Stunden, in denen sie nicht genutzt werden. Weitere Informationen finden Sie im Blogbeitrag Automatisieren Sie das Stoppen und Starten von Amazon Neptune Neptune-Umgebungsressourcen mithilfe von Ressourcen-Tags.
- Wenn Ihre Datenverkehrsmuster häufig und unvorhersehbar variieren, sollten Sie die Verwendung von Neptune Serverless-Instances in Betracht ziehen, die je nach Bedarf hoch- und herunterskalieren, anstatt eine Instanz zu verwenden, die für Spitzenverkehr bereitgestellt wurde.
- Erwägen Sie, Ihre Service Level Agreements neben den Zielen zur Geschäftskontinuität auch an Nachhaltigkeitszielen auszurichten. Durch die Vereinfachung von Anforderungen wie Disaster Recovery in mehreren Regionen, hohe Verfügbarkeit oder langfristige Aufbewahrung von Backups, insbesondere für Umgebungen außerhalb der Produktion oder für nicht geschäftskritische Workloads, kann der zur Erreichung dieser Ziele erforderliche Ressourcenaufwand reduziert werden.

Optimieren Sie Softwareentwicklung und Architekturmuster

Um Verschwendung zu vermeiden, sollten Sie Ihre Modelle und Abfragen optimieren und Rechenressourcen gemeinsam nutzen, sodass Sie alle in Neptune-Instances und -Clustern verfügbaren Ressourcen nutzen können. Zu den spezifischen bewährten Methoden gehören:

- Lassen Sie Entwickler Neptune-Instanzen und Jupyter Notebook-Anwendungsinstanzen gemeinsam nutzen, anstatt jeweils ihre eigenen zu erstellen. Geben Sie jedem Entwickler mithilfe von Multi-Tenancy-Partitionierungsstrategien seine eigene logische Partition in einem einzigen Neptune-Cluster und erstellen Sie separate Notebook-Ordner für jeden Entwickler auf einer einzigen Jupyter-Instanz.

- Implementieren Sie Muster, die den Ressourcenverbrauch maximieren und Leerlaufzeiten minimieren, z. B. parallel Threads zum Laden von Daten und zum Zusammenfügen von Datensätzen zu einer größeren Transaktion.
- Optimieren Sie Ihre Abfragen und Ihr Grafikmodell, um den Ressourcenaufwand für die Berechnung der Ergebnisse zu minimieren.
- Verwenden Sie für Gremlin-Abfrageergebnisse die Funktion zum Zwischenspeichern von Ergebnissen, um den Ressourcenaufwand für die Neuberechnung paginierter oder häufig wiederkehrender Abfragen zu minimieren.
- Halten Sie Ihre Neptun-Umgebungen auf dem neuesten Stand. Die neuesten Versionen von Neptune unterstützen die neuesten Amazon EC2 EC2-Instances wie Graviton, die effizienter sind. Sie bieten auch Verbesserungen bei der Abfrageoptimierung und Fehlerkorrekturen, die den Ressourcenaufwand für die Berechnung Ihrer Abfragen reduzieren.

Ressourcen

Referenzen

- AWS Well-Architected
- AWS Well-Architected-Framework-Dokumentation
- Die neuesten Updates von Neptune
- Best Practices: Das Beste aus Neptune herausholen
- Amazon Neptune Preisrechner

Blog-Posts

- Automatisiertes Testen des Amazon Neptune Neptune-Datenzugriffs mit Apache Gremlin TinkerPop
- Automatisieren Sie das Stoppen und Starten von Amazon Neptune Neptune-Umgebungsressourcen mithilfe von Ressourcen-Tags
- Feinkörnige Zugriffskontrolle für Amazon Neptune Neptune-Aktionen auf der Datenebene
- 4,7-mal besseres Preis-Leistungs-Verhältnis für Schreibabfragen mit AWS Graviton4 R8g-Instances, die Amazon Neptune v1.4.5 verwenden
- Wie Orca Security die Leistung seiner Amazon Neptune Neptune-Datenbank optimierte
- Schnellere Erstellung von Diagrammanwendungen mit öffentlichen Endpunkten von Amazon Neptune
- Die neue Amazon Neptune Engine-Version bietet bis zu 9-mal schnelleren und 10-mal höheren Durchsatz für OpenCypher-Abfrageleistung

AWS Kostenlose Skill Builder-Kurse

- Erste Schritte mit Amazon Neptune
- Anwendungen auf Amazon Neptune erstellen
- Datenmodellierung für Amazon Neptune

Mitwirkende

Zu den Mitwirkenden an diesem Leitfaden gehören:

- Brian O'Keefe, leitender Architekt von Neptune Solutions, AWS
- Abhishek Mishra, leitender Architekt von Neptune Solutions, AWS
- Ganesh Sawhney, Teamleiter — Architekt für Erfolgslösungen für strategische Partner, AWS
- Michael Havey, leitender Architekt von Neptune Solutions, AWS
- Kevin Phillips, Architekt von Neptune Solutions, AWS
- Melissa Kwok, Architektin von Neptune Solutions, AWS
- Sakti Mishra, Hauptarchitektin für Lösungen AWS
- Javed Ali, leitender Lösungsarchitekt, AWS

Dokumentverlauf

In der folgenden Tabelle werden wichtige Änderungen in diesem Leitfaden beschrieben. Um Benachrichtigungen über zukünftige Aktualisierungen zu erhalten, können Sie einen RSS-Feed abonnieren.

Änderung	Beschreibung	Datum
Neptun-Release-Updates	Wir haben die Dokumentation aktualisiert und enthält nun Informationen zu Amazon Neptune 1.4.6.0 und höher.	2. Januar 2026
Erste Veröffentlichung	—	27. September 2023

Die vorliegende Übersetzung wurde maschinell erstellt. Im Falle eines Konflikts oder eines Widerspruchs zwischen dieser übersetzten Fassung und der englischen Fassung (einschließlich infolge von Verzögerungen bei der Übersetzung) ist die englische Fassung maßgeblich.