



Architektur-Diagramme

Datenbank DevOps auf AWS



Datenbank DevOps auf AWS: Architektur-Diagramme

Copyright © 2026 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Die Handelsmarken und die Handelsaufmachung von Amazon dürfen nicht in einer Weise in Verbindung mit nicht von Amazon stammenden Produkten oder Services verwendet werden, durch die Kunden irregeführt werden könnten oder Amazon in schlechtem Licht dargestellt oder diskreditiert werden könnte. Alle anderen Handelsmarken, die nicht Eigentum von Amazon sind, gehören den jeweiligen Besitzern, die möglicherweise zu Amazon gehören oder nicht, mit Amazon verbunden sind oder von Amazon gesponsert werden.

Table of Contents

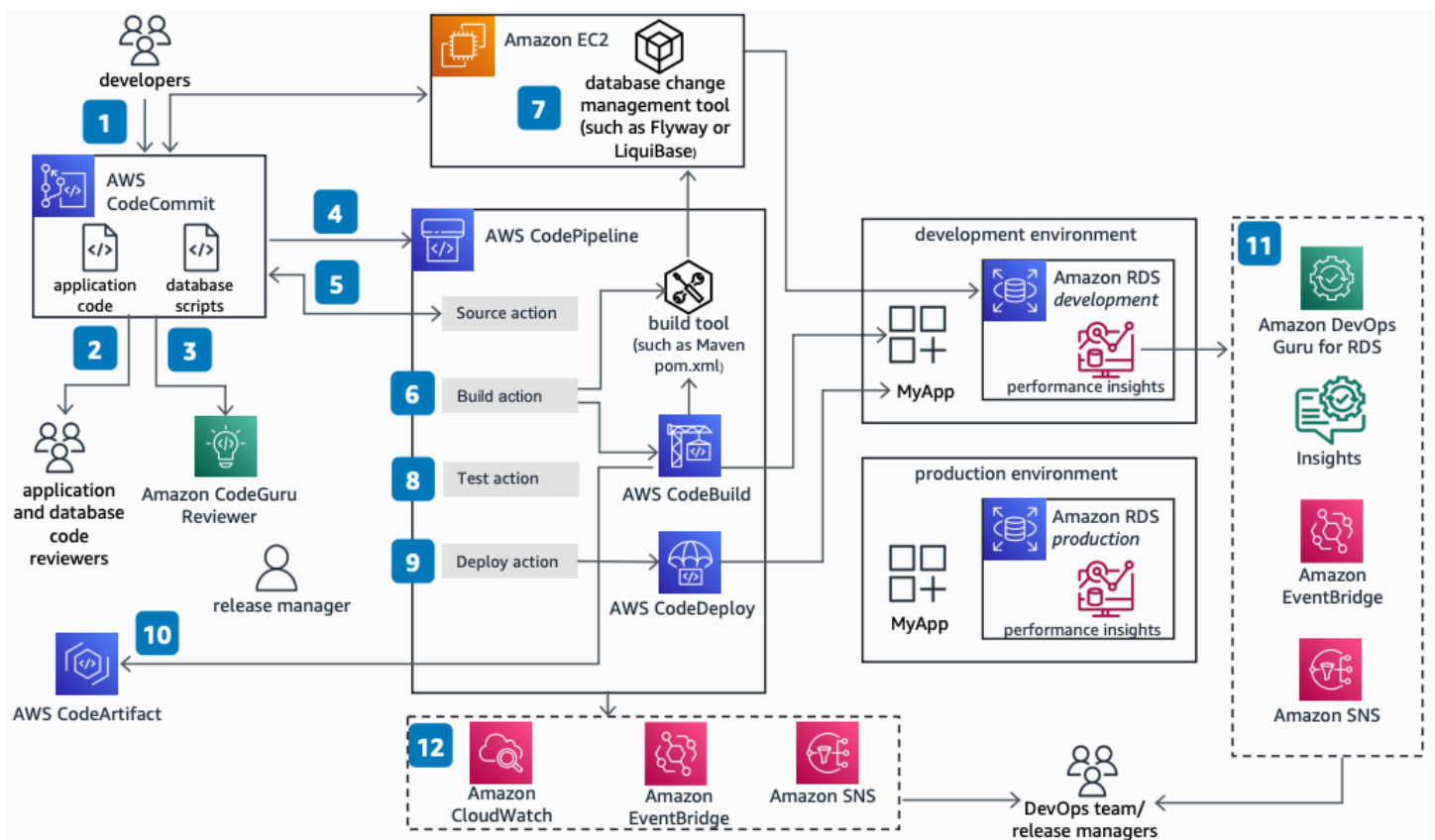
Entwicklungs-Workflow	1
Datenbank DevOps auf AWS: Entwicklungsworkflow	1
Weitere Informationen	2
Verlauf des Diagramms	2
Arbeitsablauf für die Produktion	4
Datenbank DevOps auf AWS: Produktionsablauf	4
.....	vi

Datenbank DevOps auf AWS: Entwicklungsworkflow

Veröffentlichungsdatum: 31. März 2023 ([Geschichte des Diagramms](#))

Verwenden Sie diese Architektur im Entwicklungsworkflow, um Datenbankänderungen im Rahmen der DevOps Praxis zu automatisieren. Entwickeln Sie eine Orchestrierung, um Datenbankänderungen mit der gleichen Geschwindigkeit wie Anwendungscode bereitzustellen, gemeinsame Codeüberprüfungen zwischen Datenbank und Anwendung durchzusetzen, Datenbank- und Anwendungscodeoperationen zu integrieren, ungewöhnliches Systemverhalten zu erkennen und Benachrichtigungen an beauftragte Teams zu automatisieren.

Datenbank DevOps auf AWS: Entwicklungsworkflow



Die folgenden Schritte beschreiben die Architektur:

1. Der Entwickler checkt sowohl Anwendungs- als auch Datenbankcode ein [AWS CodeCommit](#).
2. AWS CodeCommit Erzwingt bei jeder Push-Anfrage Code-Reviews mithilfe von Genehmigungsregeln.

3. Amazon CodeGuruDer Reviewer bewertet Codeänderungen automatisch, erkennt Fehler und gibt Codeempfehlungen.
4. AWS CodeCommit veranlasst, Pipeline-Maßnahmen [CodePipeline](#) zu ergreifen.
5. Die CodePipeline Quellaktion checkt Anwendungs- und Datenbankcode aus.
6. Die CodePipeline Build-Aktion ruft ein Build-Management-Tool auf, um die Erstellungsphase für Anwendungs- und Datenbankcode zu starten.
7. Das Build-Tool ruft das Tool zum Ändern der Datenbank auf. Es lädt die Datenbankskripten herunter und führt sie in der Zieldatenbank aus.
8. Das Build-Tool ruft Datenbanktestskripte auf und validiert die Ausgabe.
9. Die Aktion „ CodePipeline Bereitstellen“ stellt Code in der Zielumgebung bereit.
- 10Build-Artefakte werden in [AWS veröffentlicht, CodeArtifact](#) damit andere sie nutzen können.
- 11(Optional) Verwenden Sie Performance Insights auf [Amazon RDS](#) und Amazon DevOps Guru wenden Sie automatisch ML-Techniken an, um Leistungsengpässe und Betriebsprobleme zu erkennen.
- 12[EventBridge](#)erfasst Ereignisse und sendet sie zur Benutzerbenachrichtigung an ein [Amazon SNS](#)-Thema.

Weitere Informationen

Weitere Informationen finden Sie in den folgenden Ressourcen:

- [AWS Architektur-Ikonen](#)
- [AWS Zentrum für Architektur](#)
- [AWS Well-Architected](#)

Geschichte des Diagramms

Abonnieren Sie den RSS-Feed, um über Aktualisierungen dieses Referenzarchitekturdiagramms informiert zu werden.

Änderung

Beschreibung

Datum

Erste Veröffentlichung

Das Referenzarchitektu
rdiagramm wurde zuerst
veröffentlicht.

31. März 2023

 Note

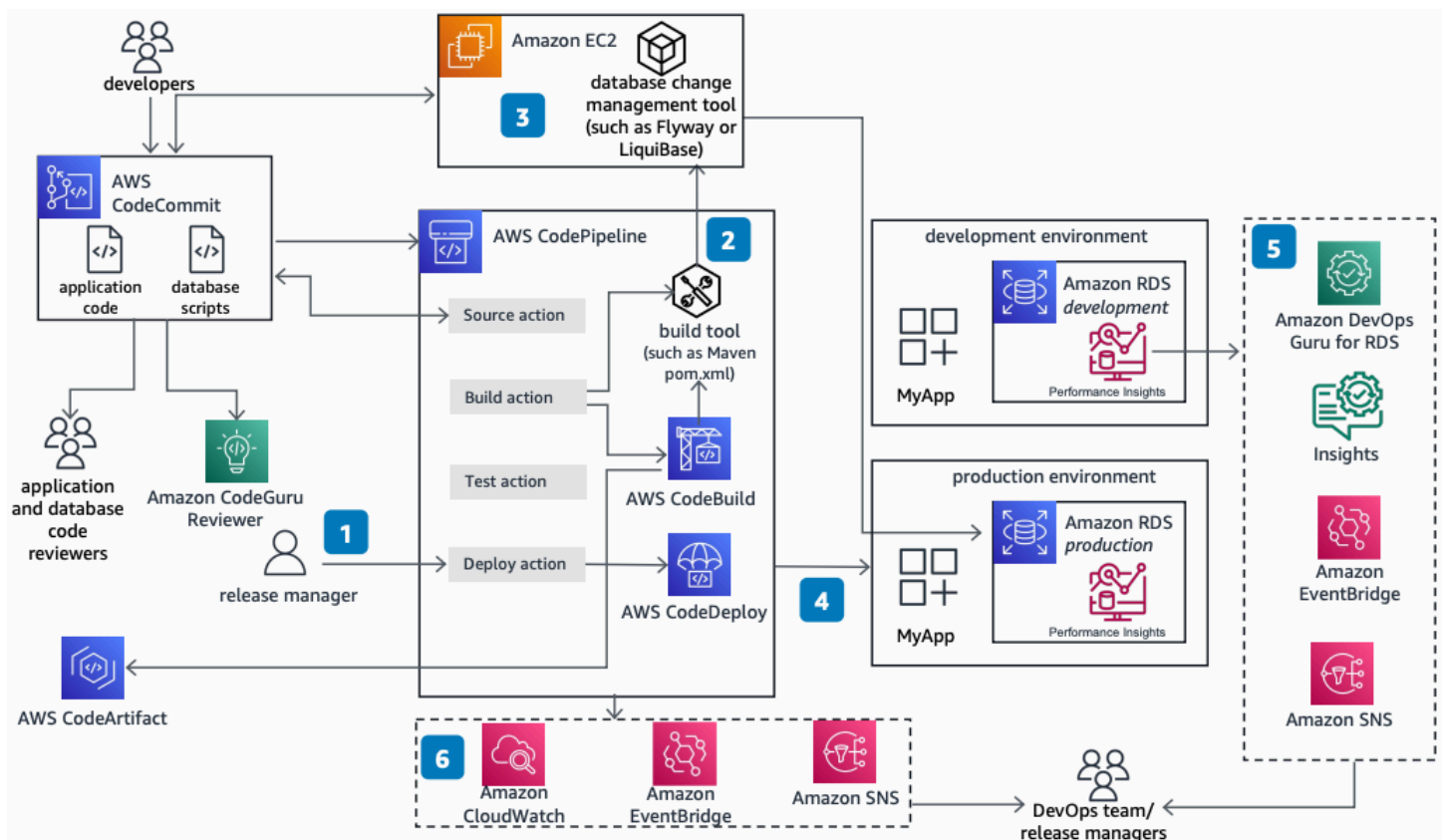
Um RSS-Updates zu abonnieren, muss ein RSS-Plugin für den von Ihnen verwendeten Browser aktiviert sein.

Datenbank DevOps auf AWS: Produktionsablauf

Veröffentlichungsdatum: 31. März 2023 ([Geschichte des Diagramms](#))

Verwenden Sie diese Architektur im Produktionsablauf, um Datenbankänderungen im Rahmen der DevOps Praxis zu automatisieren. Erstellen Sie eine Orchestrierung, um Datenbankänderungen mit der gleichen Geschwindigkeit wie der Anwendungscode bereitzustellen, ungewöhnliches Systemverhalten zu erkennen und Benachrichtigungen an beauftragte Teams zu automatisieren, um Korrekturmaßnahmen zu ergreifen.

Datenbank DevOps auf AWS: Produktionsablauf



Die folgenden Schritte beschreiben die Architektur:

1. Der Release-Manager stellt eine Anfrage zur Produktionsbereitstellung.
2. [CodePipeline](#) ruft das Build-Management-Tool auf, das das Datenbank-Change-Management-Tool aufruft.

3. Das Datenbankänderungstool lädt die Datenbankänderungsskripts herunter und führt sie in der [Amazon](#) RDS-Zielproduktionsdatenbank aus.
4. CodePipeline initiiert die Aktion Deploy, um Code in der Zielumgebung bereitzustellen.
5. (Optional) Verwenden Sie Performance Insights auf Amazon RDS und wenden Sie automatisch ML-Techniken Amazon DevOps Guru an, um Leistungsengpässe und Betriebsprobleme zu erkennen.
6. [EventBridge](#) erfasst Ereignisse und sendet sie zur Benutzerbenachrichtigung an ein [Amazon](#) SNS-Thema.

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.