



Strategi migrasi untuk database relasional

AWS Bimbingan Preskriptif



AWS Bimbingan Preskriptif: Strategi migrasi untuk database relasional

Copyright © 2024 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Merek dagang dan tampilan dagang Amazon tidak boleh digunakan sehubungan dengan produk atau layanan apa pun yang bukan milik Amazon, dengan cara apa pun yang dapat menyebabkan kebingungan di antara para pelanggan, atau dengan cara apa pun yang menghina atau mendiskreditkan Amazon. Semua merek dagang lain yang tidak dimiliki oleh Amazon merupakan properti dari masing-masing pemilik, yang mungkin berafiliasi, terkait dengan, atau disponsori oleh Amazon, atau tidak.

Table of Contents

Pengantar	1
Gambaran Umum	1
Fase migrasi database	3
Tahap 1: Mempersiapkan	4
Mengidentifikasi dependensi	4
Memenuhi beban kerja	5
Tahap 2: Rencana	7
Pilih strategi migrasi	8
Tahap 3: Migrasi	10
Mengubah skema	10
AWS SCT	11
Playbook migrasi	12
Migrasi data	12
AWS DMS	12
Opsi migrasi offline	13
Memperbarui aplikasi	13
Menguji migrasi	14
Potong	14
Migrasi offline	14
Migrasi Flash-cut	15
Konfigurasi database aktif/aktif	15
Migrasi tambahan	16
Ikuti praktik terbaik diAWS	16
Tahap 4: Mengoperasikan dan mengoptimalkan	17
MenggunakanAWSMitra	19
Langkah selanjutnya	20
Sumber daya	21
Riwayat dokumen	22
Glosarium	23
#	23
A	24
B	27
C	28
D	31

E	35
F	37
G	38
H	39
I	40
L	43
M	44
O	47
P	49
Q	52
R	52
D	55
T	58
U	60
V	60
W	61
Z	62
.....	lxiii

Strategi migrasi untuk database relasional

Yaser King, Amazon Web Services (AWS)

November 2020([riwayat dokumen](#))

Dalam portofolio perusahaan Anda, Anda cenderung memiliki beberapa jenis database. Saat Anda bermigrasi ke Amazon Web Services (AWS), Anda dapat memilih untuk melakukan “mengangkat dan menggeser” database Anda (rehost) atau memodernisasi aplikasi Anda dengan beralih keAWSlayanan database terkelola (replatform).

Jika Anda memilih untuk rehost database Anda,AWSmenyediakan sejumlah layanan dan alat yang dapat membantu Anda memindahkan, menyimpan, dan menganalisis data Anda dengan aman. Jika Anda memilih untuk beralih keAWSbasis data terkelola,AWSmenawarkan banyak pilihan sehingga Anda tidak perlu menukar fungsionalitas, kinerja, atau skala. Untuk informasi lebih lanjut tentangAWSkeluarga database, lihat[Basis data diAWS](#)padaAWSwebsite.

Dokumen ini berfokus pada strategi untuk memigrasi database relasional keAWSCloud, untuk eksekutif TI dan bisnis, manajer program atau proyek, pemilik produk, dan manajer operasi/ infrastruktur yang berencana memigrasi database lokal mereka keAWS.

Gambaran Umum

Strategi migrasi database terbaik memungkinkan Anda memanfaatkan sepenuhnyaAWSCloud. Ini melibatkan migrasi aplikasi Anda untuk menggunakan database cloud-native yang dibuat khusus. Anda tidak boleh membatasi diri Anda pada database penjaga lama yang sama yang telah Anda gunakan di lokasi. Sebagai gantinya, pertimbangkan untuk memodernisasi aplikasi Anda dan pilih database yang paling sesuai dengan persyaratan alur kerja aplikasi Anda.

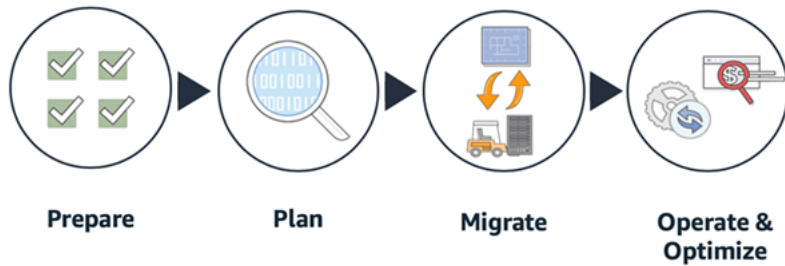
Banyak perusahaan telah mengadopsi pendekatan ini. Misalnya, Airbnb perlu memproses dan menganalisis 50 GB data dengan cepat setiap hari. Mereka membutuhkan basis data nilai kunci untuk menyimpan riwayat pencarian pengguna untuk pencarian cepat yang memungkinkan pencarian yang dipersonalisasi, penyimpanan data dalam memori untuk menyimpan status sesi untuk rendering situs (sub-milidetik) yang lebih cepat, dan database relasional sebagai basis data transaksional utama mereka. Mereka memilih[Amazon DynamoDB](#)sebagai basis data kunci-nilai mereka,[Amazon ElastiCache](#)sebagai toko dalam memori mereka, dan[Amazon Relational Database Service](#)(Amazon RDS) untuk basis data transaksional mereka. Untuk informasi selengkapnya tentang cara AirbnbAWSlayanan database, lihat[Studi kasus Airbnb](#).

Strategi migrasi database terkait erat dengan strategi cloud menyeluruh organisasi Anda. Misalnya, jika Anda memilih untuk pertama transisi aplikasi Anda dan kemudian mengubah mereka, Anda mungkin memutuskan untuk mengangkat dan menggeser database Anda terlebih dahulu. Ketika Anda sepenuhnya berada diAWS Cloud, Anda mulai bekerja untuk memodernisasi aplikasi Anda. Strategi ini dapat membantu Anda keluar dari pusat data Anda saat ini dengan cepat, dan kemudian fokus pada modernisasi.

Migrasi database Anda digabungkan erat dengan migrasi aplikasi Anda. Semua strategi migrasi database melibatkan beberapa tingkat perubahan pada aplikasi yang menggunakan database tersebut. Perubahan ini berkisar dari menunjuk ke lokasi baru database diAWS Cloud ke total penulisan ulang aplikasi, jika tidak dapat diubah karena kode sumber tidak tersedia, atau merupakan aplikasi pihak ketiga sumber tertutup.

Fase migrasi database

Ketika Anda telah mengidentifikasi database untuk migrasi, Anda akan melalui fase persiapan, perencanaan, migrasi, dan optimalisasi database.



Bagian berikut membahas setiap fase secara rinci:

- [Fase 1: Siapkan](#)
- [Tahap 2: Rencana](#)
- [Fase 3: Migrasi](#)
- [Fase 4: Operasikan dan optimalkan](#)

Tahap 1: Mempersiapkan

Tahap pertama dari proses migrasi database adalah persiapan. Selama persiapan, Anda mengidentifikasi interdependensi antara aplikasi dan database Anda. Anda juga menganalisis beban kerja database untuk menentukan kategori migrasi: dari migrasi rehost sederhana (homogen) hingga migrasi arsitek ulang (heterogen). Tanpa menyelesaikan fase ini, Anda berisiko mengalami jadwal migrasi yang tertunda.

Tugas ini dibahas dalam bagian berikut:

- [Mengidentifikasi dependensi](#)
- [Beban kerja yang memenuhi syarat](#)

Mengidentifikasi dependensi

Anda mulai dengan mengidentifikasi dependensi aplikasi dan database, dengan mengajukan pertanyaan seperti berikut:

- Apakah database ini langsung diakses oleh aplikasi lain?

Jika demikian, Anda harus menentukan bagaimana migrasi database mempengaruhi aplikasi itu. Jika Anda rehosting database, Anda perlu memastikan bahwa aplikasi masih dapat mengakses database dengan kinerja yang dapat diterima.

- Apakah aplikasi langsung mengakses database lain?

Jika demikian, tentukan rencana migrasi untuk database lainnya. Jika itu juga bermigrasi, Anda perlu memperbarui aplikasi yang sesuai. Jika tidak bermigrasi, Anda perlu memastikan bahwa aplikasi dapat terus terhubung dengan latensi yang dapat diterima.

- Apakah database menggunakan link database untuk mengambil data dari database lain?

Seperti pada titik sebelumnya, tentukan rencana migrasi untuk database lain dan tangani tautan yang sesuai.

- Apakah aplikasi tergantung pada perangkat lunak lokal apa pun?

Jika demikian, Anda harus menentukan rencana migrasi untuk perangkat lunak tersebut. Jika itu bermigrasi, Anda perlu memperbarui aplikasi Anda sesuai. Jika tidak, pastikan bahwa aplikasi dapat terus terhubung ke perangkat lunak dan latensi dapat diterima.

- Apakah ada dependensi perangkat keras?

Jika demikian, datang dengan rencana untuk mengatasi mereka.

- Apakah ada persyaratan bandwidth atau jaringan yang ketat?

Jika demikian, pilih AWS layanan yang dapat membantu Anda memenuhi persyaratan ini.

- Apakah aplikasi menggunakan opsi atau fitur mesin database khusus?

Jika Anda bermigrasi ke mesin database yang berbeda, Anda perlu memperbarui aplikasi yang sesuai.

Jika jawaban atas pertanyaan-pertanyaan ini rumit, pilihan yang lebih baik adalah memisahkan database dari aplikasi dengan menggunakan layanan mikro. Dengan cara ini, aplikasi bisa mendapatkan data dengan memanggil microservice alih-alih langsung terhubung ke database.

Memenuhi beban kerja

Untuk menentukan strategi migrasi terbaik untuk database Anda, penting untuk memahami beban kerja database saat ini. Anda perlu menganalisis database Anda untuk menentukan fitur mana yang sedang Anda gunakan dan apa yang terlibat dalam migrasi ke mesin database cloud-native lain seperti [Amazon Aurora PostgreSQL](#).

AWS menyediakan alat kualifikasi beban kerja yang disebut AWS Kerangka Kualifikasi Beban Kerja (AWS WQF). Alat ini dapat membantu mengidentifikasi kompleksitas migrasi database Oracle dan Microsoft SQL Server Anda dengan menganalisis skema database dan objek kode, kode aplikasi, dependensi, karakteristik kinerja, dan input serupa. WQF memberikan rekomendasi pada mesin database target. Hal ini juga memperkirakan jenis pekerjaan yang terlibat dan tingkat usaha yang diperlukan.

WQF mengevaluasi beban kerja migrasi Anda dan menempatkannya di salah satu dari lima kategori beban kerja, diringkas dalam tabel berikut.

	Category 1	ODBC/JDBC workloads	< 50 manual changes, easy to refactor
	Category 2	Light, proprietary feature workloads	< 200 manual changes, medium complexity
	Category 3	Heavy, proprietary feature workloads	> 200 manual changes, high complexity
	Category 4	Engine-specific workloads	Not recommended for refactoring
	Category 5	COTS or other non-portable workloads	Not recommended for refactoring

- Kategori 1: Beban kerja yang menggunakan Open Database Connectivity (ODBC) atau Java Database Connectivity (JDBC) bukan driver eksklusif untuk terhubung ke basis data. Kategori ini biasanya memiliki prosedur tersimpan sederhana yang digunakan untuk kontrol akses. Konversi membutuhkan kurang dari 50 perubahan manual.
- Kategori 2: Beban kerja dengan penggunaan ringan fitur eksklusif dan yang tidak menggunakan fitur bahasa SQL tingkat lanjut. Jenis beban kerja ini membutuhkan kurang dari 200 perubahan manual.
- Kategori 3: Beban kerja dengan penggunaan fitur eksklusif yang berat. Beban kerja dalam kategori ini benar-benar didorong oleh logika prosedur tersimpan canggih atau fitur eksklusif. Jenis beban kerja ini membutuhkan lebih dari 200 perubahan manual yang melibatkan kode dan fitur database-resident.
- Kategori 4: Beban kerja khusus mesin. Beban kerja dalam kategori ini menggunakan kerangka kerja yang hanya dapat bekerja dengan mesin database komersial tertentu. Misalnya, kerangka kerja ini mungkin termasuk Oracle Forms, Oracle Reports, Oracle Application Development Framework (ADF), Oracle Application Express (APEX), atau aplikasi yang menggunakan NET ActiveRecord secara ekstensif.
- Kategori 5: Beban kerja yang tidak portabel, tidak dapat diterima, atau “angkat dan pergeseran”. Beban kerja dalam kategori ini mungkin diimplementasikan pada mesin database yang tidak memiliki ekuivalen berbasis cloud. Dalam beberapa kasus, Anda mungkin tidak memiliki kode sumber untuk program ini.

Kategorisasi ini dapat membantu Anda menentukan jalur migrasi untuk aplikasi Anda, seperti yang akan kita bahas di bagian [Tahap 2: Paket](#).

AWS saat ini tidak menyediakan AWS WQF untuk men-download. Jika Anda memerlukan bantuan untuk menilai migrasi ke AWS bersama AWS WQF, kami sarankan membuka tiket dukungan. AWS akan terlibat dengan Anda secara langsung untuk membantu membuat proses bekerja untuk Anda.

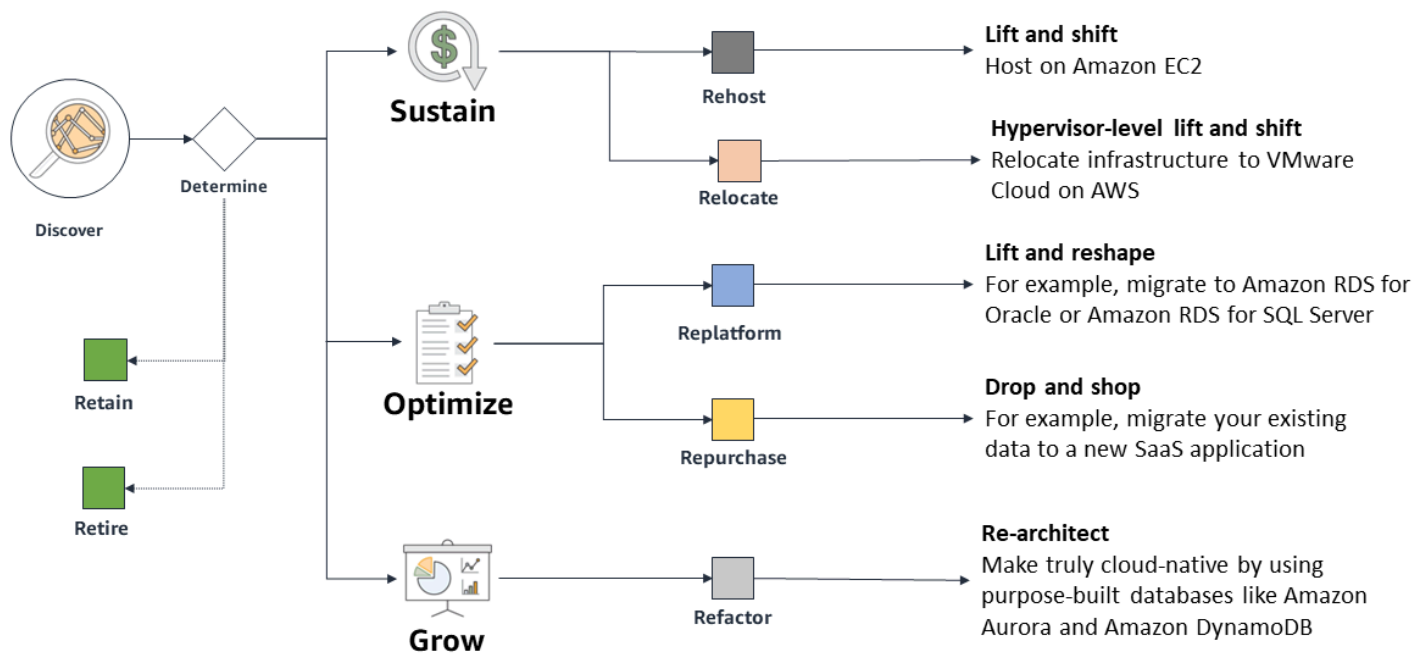
Tahap 2: Rencana

Pada fase ini, Anda menggunakan informasi yang dikumpulkan selama fase persiapan dan menghasilkan strategi migrasi. Aspek penting dari perencanaan migrasi adalah merasionalisasi informasi yang Anda kumpulkan terhadap 7 Rs migrasi: rehost, replatform, relokasi, pembelian kembali, refactor, pensiun, dan pertahankan.

Memilih strategi migrasi Anda bergantung pada driver bisnis Anda untuk adopsi cloud, serta pertimbangan waktu, kendala bisnis dan keuangan, dan persyaratan sumber daya. Jika Anda ingin mempertahankan beban kerja Anda saat ini di cloud, pilih rehosting. Namun, jika Anda ingin mengoptimalkan dan menskalakan beban kerja Anda, pertimbangkan salah satu opsi lainnya.

Berikut adalah ikhtisar dari 7 Rs migrasi database. Ini diilustrasikan dalam diagram berikut.

Migration Paths



- Rehost (angkat dan geser) — Pindahkan aplikasi ke cloud tanpa membuat perubahan apa pun. Misalnya, memigrasikan database Oracle lokal Anda ke Oracle di instans Amazon [Elastic Compute Cloud \(Amazon EC2\)](#) di Cloud. AWS
- Relokasi (hypervisor-level lift and shift) — Pindahkan infrastruktur ke cloud tanpa membeli perangkat keras baru, menulis ulang aplikasi, atau memodifikasi operasi yang ada. Skenario migrasi ini khusus untuk VMware Cloud on AWS, yang mendukung kompatibilitas mesin virtual (VM) dan portabilitas beban kerja antara lingkungan lokal Anda dan AWS. Anda dapat

menggunakan teknologi VMware Cloud Foundation dari pusat data lokal saat memigrasikan infrastruktur ke VMware Cloud. AWS Misalnya, pindahkan hypervisor yang menghosting database Oracle Anda ke VMware Cloud on. AWS






- Replatform (angkat dan bentuk ulang) — Pindahkan aplikasi ke cloud, dan perkenalkan beberapa tingkat pengoptimalan untuk memanfaatkan kemampuan cloud. Misalnya, memigrasikan database Oracle lokal Anda ke [Amazon RDS for Oracle di Cloud](#). AWS
- Pembelian kembali (drop and shop) — Ubah ke produk yang berbeda, biasanya dengan berpindah dari aplikasi tradisional ke produk perangkat lunak sebagai layanan (SaaS), dan memigrasikan data dari aplikasi lokal Anda ke produk baru. Misalnya, memigrasikan data pelanggan Anda dari sistem manajemen hubungan pelanggan (CRM) lokal Anda ke Salesforce.com.
- Refactor (re-architect) - Memindahkan aplikasi dan memodifikasi arsitekturnya dengan memanfaatkan sepenuhnya fitur cloud-native untuk meningkatkan kelincahan, kinerja, dan skalabilitas. [Misalnya, memigrasikan database Oracle lokal Anda ke Aurora PostgreSQL](#). Strategi ini juga dapat mencakup penulisan ulang aplikasi Anda untuk menggunakan basis data yang dibuat khusus yang menawarkan alur kerja yang AWS berbeda. Atau, Anda dapat memilih untuk memodernisasi aplikasi monolitik Anda dengan memecahnya menjadi layanan mikro yang lebih kecil yang mengakses skema database mereka sendiri.
- Pertahankan (kunjungi kembali) - Simpan aplikasi di lingkungan sumber Anda. Ini mungkin termasuk aplikasi yang memerlukan refactoring besar, dan Anda ingin menunda pekerjaan itu sampai nanti, dan aplikasi lama yang ingin Anda pertahankan karena tidak ada pembenaran bisnis untuk memigrasikannya.
- Pensiun — Menonaktifkan atau menghapus aplikasi yang tidak lagi diperlukan di lingkungan sumber Anda.

Pilih strategi migrasi

Di sebagian besar migrasi database, Anda dapat memilih untuk meng-host ulang, memplatform ulang, atau memfaktorkan ulang. Salah satu strategi ini dapat bekerja untuk Anda. Prinsip panduannya adalah bagaimana Anda bisa mendapatkan manfaat maksimal dari migrasi Anda. Memilih untuk memfaktorkan ulang aplikasi Anda dan bermigrasi ke database cloud-native seperti Aurora dapat memungkinkan Anda untuk meningkatkan aplikasi database Anda. Namun, tergantung pada kompleksitas beban kerja Anda, refactoring database dapat memakan waktu dan sumber daya intensif.

Kategorisasi WQF membantu Anda memutuskan kapan Anda harus mempertimbangkan strategi migrasi tertentu. Kategori WQF yang lebih tinggi berarti bahwa upaya migrasi yang diperlukan adalah

signifikan; oleh karena itu, Anda mungkin ingin memilih opsi lain, seperti rehost atau replatform, untuk menyelesaikan migrasi dalam jangka waktu yang dapat diterima. Tabel berikut menunjukkan strategi yang disarankan berdasarkan kategori WQF.

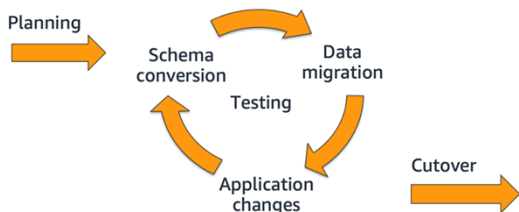
Kategori	Kompleksitas beban kerja	Beban kerja	Strategi migrasi
1		Beban kerja ODBC/JBDC	Kandidat untuk refactor
2		Beban kerja fitur yang ringan dan eksklusif	Kandidat untuk refactor
3		Beban kerja fitur yang berat dan eksklusif	Kandidat untuk refactor atau replatform
4		Beban kerja khusus mesin	Kandidat untuk replatform atau rehost
5		Non-portabel, berisiko tinggi, atau beban kerja lift-and-shift	Kandidat untuk replatform atau rehost

Opsi rehost dan replatform cocok ketika kompleksitas yang terlibat dalam refactoring tinggi.

Dalam skenario ini, berdasarkan kebutuhan modernisasi, Anda dapat mempertimbangkan untuk memfaktorkan ulang database setelah menyelesaikan migrasi ke Cloud. AWS

Tahap 3: Migrasi

Setelah Anda menyelesaikan perencanaan migrasi dan mengidentifikasi strategi migrasi, migrasi aktual terjadi. Pada fase ini, database target dirancang, data sumber dimigrasi ke target, dan data divalidasi.



Ini adalah proses berulang yang mencakup beberapa siklus konversi, migrasi, dan pengujian. Setelah pengujian fungsional dan kinerja selesai, Anda dapat memotong ke database baru.

Fase migrasi terdiri dari langkah-langkah kunci berikut, yang dibahas di bagian berikut:

- [Mengubah skema](#)
- [Migrasi data](#)
- [Memperbarui aplikasi](#)
- [Menguji migrasi](#)
- [Memotong ke database baru](#)

Mengubah skema

Salah satu tugas utama selama migrasi database adalah untuk memigrasikan skema Anda dari mesin database sumber ke mesin database target. Jika Anda rehost atau replatform, mesin database Anda tidak akan berubah. Ini disebut sebagaimigrasi database homogen, dan Anda dapat menggunakan alat database asli Anda untuk memigrasikan skema.

Namun, jika Anda merancang ulang aplikasi Anda, konversi skema mungkin memerlukan lebih banyak usaha. Dalam hal ini, Anda akan melakukan migrasi database heterogen, di mana sumber dan target mesin database Anda akan berbeda. Skema database Anda saat ini mungkin menggunakan paket dan fitur yang tidak dapat langsung dikonversi ke mesin database target. Beberapa fitur mungkin tersedia dengan nama yang berbeda. Oleh karena itu, mengkonversi skema

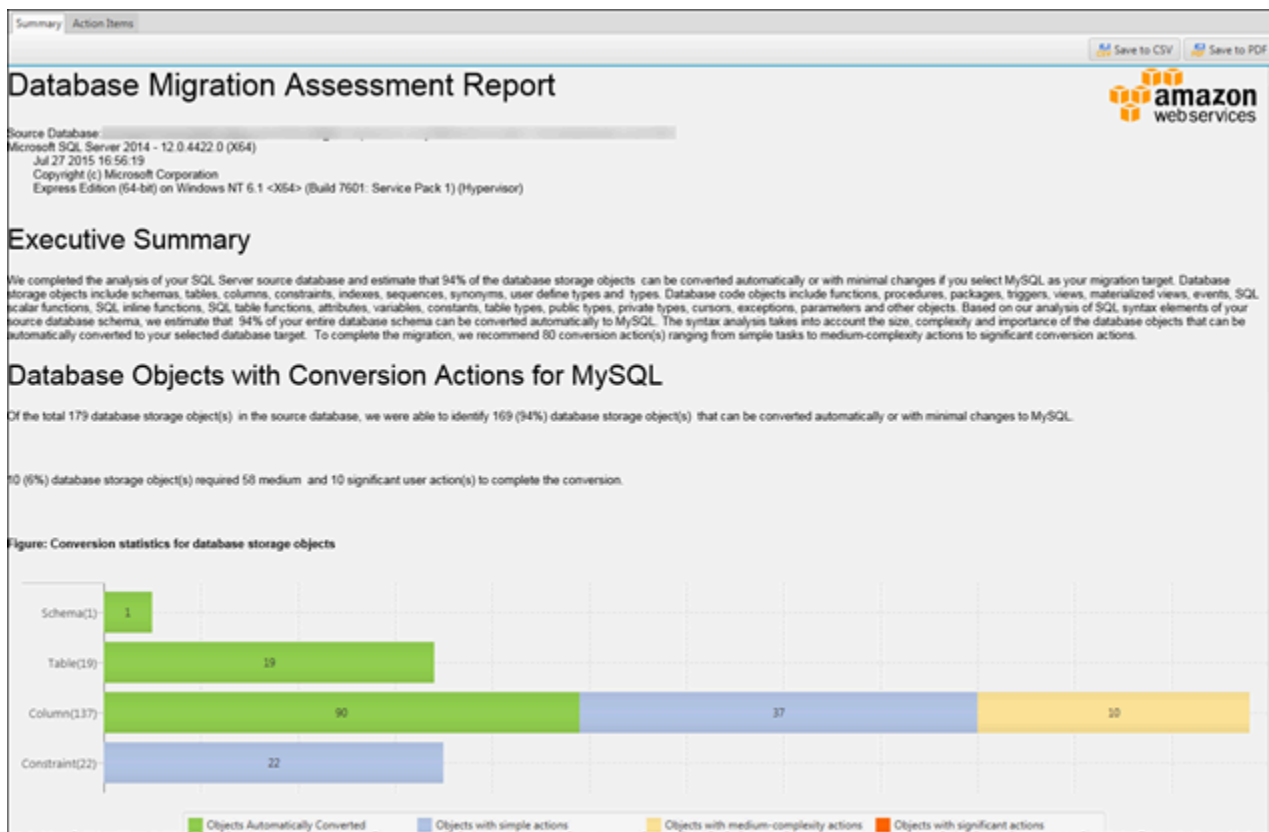
mempunyai pemahaman yang baik dari sumber dan target mesin database Anda. Tugas ini bisa menantang, tergantung pada kompleksitas skema Anda saat ini.

AWS menyediakan dua sumber daya untuk membantu Anda dengan konversi skema: AWS Schema Conversion Tool (AWS SCT) dan pedoman migrasi.

AWS SCT

AWS SCT adalah alat gratis yang dapat membantu Anda mengkonversi database yang ada dari satu mesin ke mesin lain. AWS SCT mendukung sejumlah basis data sumber, termasuk Oracle, MySQL, Sybase, dan IBM Db2 LUW. Anda dapat memilih basis data target seperti Aurora MySQL dan Aurora PostgreSQL.

AWS SCT menyediakan antarmuka pengguna grafis yang langsung terhubung ke sumber dan target database untuk mengambil objek skema saat ini. Ketika terhubung, Anda dapat membuat laporan penilaian migrasi database untuk mendapatkan ringkasan tingkat tinggi dari upaya konversi dan item tindakan. Ilustrasi layar berikut menunjukkan laporan penilaian migrasi database sampel.



Dengan AWS SCT Anda dapat mengkonversi skema dan menyebarkan ke database target secara langsung, atau Anda bisa mendapatkan file SQL untuk skema dikonversi. Untuk

informasi selengkapnya, lihat [Menggunakan AWS Schema Conversion Tool Antarmuka pengguna di AWS dokumentasi](#).

Playbook migrasi

Meskipun AWS SCT mengkonversi banyak objek sumber Anda, beberapa aspek konversi memerlukan intervensi manual dan penyesuaian. Untuk membantu tugas ini, AWS menyediakan playbook migrasi yang detail ketidakcocokan dan kesamaan antara dua database. Untuk informasi selengkapnya tentang playbook ini, lihat [AWS Database Migration Services sumber daya](#) pada AWS situs web.

Migrasi data

Ketika migrasi skema selesai, Anda dapat memindahkan data dari basis data sumber ke basis data target. Tergantung pada persyaratan ketersediaan aplikasi Anda, Anda dapat menjalankan pekerjaan ekstraksi sederhana yang melakukan salinan satu kali dari data sumber ke dalam database baru. Atau, Anda dapat menggunakan alat yang menyalin data saat ini dan terus meniru semua perubahan sampai Anda siap untuk memotong ke database baru. Untuk migrasi rehost dan replatform, sebaiknya gunakan alat khusus database asli untuk memigrasi data Anda.

Alat yang dapat membantu Anda dengan transfer data termasuk AWS Database Migration Service (AWS DMS) dan alat migrasi offline. Ini dijelaskan di bagian berikut.

AWS DMS

Setelah Anda menggunakan AWS SCT untuk mengkonversi objek skema Anda dari mesin database sumber ke mesin target, Anda dapat menggunakan AWS DMS untuk memigrasikan data. Dengan AWS DMS Anda dapat menjaga database sumber dan berjalan saat data sedang direplikasi. Anda dapat melakukan salinan satu kali data Anda atau menyalin dengan replikasi berkelanjutan. Ketika database sumber dan target disinkronkan, Anda dapat mengambil database Anda secara offline dan memindahkan operasi Anda ke database target. AWS DMS dapat digunakan untuk migrasi database homogen (misalnya, dari database Oracle lokal ke database Amazon RDS for Oracle) serta migrasi heterogen (misalnya, dari database Oracle lokal ke Amazon RDS untuk database PostgreSQL). Untuk informasi lebih lanjut tentang bekerja dengan AWS DMS, lihat [AWS DMS dokumentasi](#).

Opsi migrasi offline

Anda dapat menggunakan opsi lain selain AWS DMS untuk mengekstrak data Anda dari basis data sumber dan memuatnya ke basis data target. Opsi ini sebagian besar cocok ketika downtime aplikasi diperbolehkan selama aktivitas migrasi data. Contoh metode ini meliputi:

- Sebuah nilai koma-terpisah (CSV) ekstrak dari database sumber dimuat ke database target
- Untuk database sumber Oracle, ora2pgutilitas untuk menyalin data ke PostgreSQL
- Custom extract, transform, load (ETL) pekerjaan untuk menyalin data dari sumber ke target

Memperbarui aplikasi

Migrasi database hampir tidak pernah migrasi database-only. Anda harus melihat aplikasi yang menggunakan database untuk memastikan bahwa ia bekerja seperti yang diharapkan dengan database baru. Perubahan minimal jika Anda hanya rehosting atau replatforming mesin database yang sama, tetapi dapat lebih signifikan jika Anda memutuskan untuk pindah ke mesin database baru.

Jika aplikasi Anda bergantung pada pemetaan objek-relasional (ORM) untuk berinteraksi dengan database, aplikasi Anda tidak akan memerlukan banyak perubahan ketika Anda bermigrasi ke mesin database baru. Namun, jika aplikasi Anda memiliki interaksi database khusus atau kueri SQL yang dibangun secara dinamis, perubahannya bisa cukup besar. Mungkin ada perbedaan dalam format query yang perlu diperbaiki untuk memastikan bahwa aplikasi bekerja seperti yang diharapkan.

Misalnya, di Oracle, menggabungkan string dengan NULL mengembalikan string asli. Namun, di PostgreSQL, menggabungkan string dengan NULL pulang NULL. Contoh lainnya adalah bagaimana NULL dan string kosong diperlakukan. Di PostgreSQL, NULL dan string kosong adalah dua hal yang berbeda, sedangkan database seperti Oracle memperlakukan mereka dengan cara yang sama. Di Oracle, jika Anda memasukkan baris dengan nilai kolom diatur ke NULL atau string kosong, Anda dapat mengambil kedua jenis nilai dengan menggunakan `where` Klausul: `where <mycolumn> is NULL`. Di PostgreSQL, ini `where` klausa akan kembali hanya satu baris di mana nilai kolom sebenarnya NULL; itu tidak akan mengembalikan baris yang memiliki nilai string kosong. Untuk informasi selengkapnya tentang perbedaan ini, lihat [playbook migrasi yang tercantum di AWS Database Migration Service sumber daya](#) halaman web.

Menguji migrasi

Fungsional dan pengujian kinerja adalah bagian penting dari migrasi database. Pengujian fungsional terperinci akan memastikan bahwa aplikasi Anda bekerja dengan database baru tanpa masalah. Anda harus menginvestasikan waktu untuk mengembangkan unit test untuk menguji alur kerja aplikasi.

Pengujian kinerja memastikan bahwa waktu respons database Anda berada dalam rentang waktu yang dapat diterima. Anda dapat mengidentifikasi kemacetan, mengoptimalkan, dan mengulangi tes kinerja. Anda mengulangi siklus seperti yang diperlukan untuk mendapatkan hasil kinerja yang diinginkan.

Pengujian bisa manual atau otomatis. Sebaiknya Anda menggunakan kerangka kerja otomatis untuk pengujian. Selama migrasi, Anda akan perlu untuk menjalankan tes beberapa kali, sehingga memiliki kerangka pengujian otomatis membantu mempercepat perbaikan bug dan siklus optimasi.

Pengujian ini dapat mengungkapkan masalah yang terlewatkan selama fase pengembangan. Misalnya, setiap kueri yang salah dikonversi akan gagal atau mengembalikan hasil yang salah, menyebabkan pengujian fungsional gagal. pengujian kinerja dapat mengungkapkan masalah seperti indeks hilang menyebabkan waktu respon query lambat. Mereka juga dapat mengungkapkan masalah kinerja yang memerlukan penyetelan mesin database, tergantung pada beban kerja, atau memodifikasi kueri.

Potong

Strategi cutover database biasanya erat ditambah dengan persyaratan downtime untuk aplikasi. Strategi yang dapat Anda gunakan untuk cutover database meliputi migrasi offline, migrasi flash-cut, konfigurasi database aktif/aktif, dan migrasi tambahan. Ini dibahas di bagian berikut.

Migrasi offline

Jika Anda dapat mengambil aplikasi Anda secara offline untuk jangka waktu lama selama operasi tulis, Anda dapat menggunakan AWS DMS setelah tugas beban penuh atau salah satu opsi migrasi offline untuk migrasi data Anda. Lalu lintas baca dapat berlanjut saat migrasi ini sedang berlangsung, namun lalu lintas tulis harus dihentikan. Karena semua data perlu disalin dari database sumber, sumber sumber database seperti I/O dan CPU dimanfaatkan.

Pada tingkat tinggi, migrasi offline melibatkan langkah-langkah berikut:

1. Lengkapi konversi skema.
2. Mulai downtime untuk lalu lintas tulis.
3. Migrasikan data menggunakan salah satu opsi migrasi offline.
4. Verifikasi data Anda.
5. Arahkan aplikasi Anda ke basis data baru.
6. Akhiri downtime aplikasi.

Migrasi Flash-cut

Dalam migrasi flash-cut, tujuan utamanya adalah menjaga downtime seminimal mungkin. Strategi ini bergantung pada replikasi data (CDC) dari basis data sumber ke basis data target. Semua lalu lintas baca/tulis akan berlanjut pada database saat ini saat data sedang dimigrasi. Karena semua data perlu disalin dari database sumber, sumber sumber server seperti I/O dan CPU dimanfaatkan. Anda harus menguji untuk memastikan bahwa aktivitas migrasi data ini tidak memengaruhi SLA kinerja aplikasi Anda.

Pada tingkat tinggi, migrasi flash-cut melibatkan langkah-langkah berikut:

1. Lengkapi konversi skema.
2. Mengatur AWS DMS dalam mode replikasi data terus menerus.
3. Saat database sumber dan target disinkronkan, verifikasi data.
4. Mulai downtime aplikasi.
5. Menggelar versi baru dari aplikasi, yang menunjuk ke database baru.
6. Akhiri downtime aplikasi.

Konfigurasi database aktif/aktif

Konfigurasi database aktif/aktif melibatkan pengaturan mekanisme untuk menjaga sumber dan target database sinkron sementara kedua database sedang digunakan untuk lalu lintas tulis. Strategi ini melibatkan lebih banyak pekerjaan daripada migrasi offline atau flash-cut, tetapi juga memberikan fleksibilitas lebih selama migrasi. Misalnya, selain mengalami downtime minimal selama migrasi, Anda dapat memindahkan lalu lintas produksi ke database baru dalam batch kecil yang terkontrol alih-alih melakukan cutover satu kali. Anda dapat melakukan operasi penulisan ganda sehingga perubahan dilakukan pada kedua database, atau menggunakan alat replikasi bi-directional seperti [HVR](#) untuk menjaga database tetap sinkron. Strategi ini memiliki kompleksitas yang lebih tinggi

dalam hal pengaturan dan pemeliharaan, sehingga pengujian lebih diperlukan untuk menghindari masalah konsistensi data.

Pada tingkat tinggi, konfigurasi database aktif/aktif melibatkan langkah-langkah berikut:

1. Lengkapi konversi skema.
2. Salin data yang ada dari database sumber ke database target, dan kemudian menjaga dua database sinkron dengan menggunakan alat replikasi bi-directional atau ganda menulis dari aplikasi.
3. Saat database sumber dan target disinkronkan, verifikasi data.
4. Mulai memindahkan bagian dari lalu lintas Anda ke database baru.
5. Terus bergerak lalu lintas sampai semua lalu lintas database Anda telah dipindahkan ke database baru.

Migrasi tambahan

Dalam migrasi tambahan, Anda memigrasikan aplikasi Anda di bagian yang lebih kecil daripada melakukan satu kali, cutover penuh. Strategi cutover ini bisa memiliki banyak variasi, berdasarkan arsitektur aplikasi Anda saat ini atau refactoring yang ingin Anda lakukan dalam aplikasi.

Anda dapat menggunakan [pola desain](#) untuk menambahkan layanan mikro independen baru untuk mengganti bagian dari aplikasi warisan monolitik yang ada. Layanan mikro independen ini memiliki tabel mereka sendiri yang tidak dibagi atau diakses oleh bagian lain dari aplikasi. Anda memigrasikan microservices ini ke database baru satu per satu, menggunakan salah satu strategi cutover lainnya. Layanan mikro yang dimigrasi mulai menggunakan database baru untuk lalu lintas baca/tulis sementara semua bagian lain dari aplikasi terus menggunakan database lama. Ketika semua layanan mikro telah dimigrasi, Anda mendekomisi aplikasi warisan Anda. Strategi ini memecah migrasi menjadi potongan-potongan yang lebih kecil dan dapat dikelola dan dapat, oleh karena itu, mengurangi risiko yang terkait dengan satu migrasi besar.

Ikuti praktik terbaik diAWS

Selain aktivitas migrasi yang dibahas di bagian sebelumnya, Anda harus menginvestasikan waktu untuk memastikan bahwa Anda mengikuti praktik terbaik untuk meng-host database Anda diAWS Cloud. Lihat [AWS dokumentasi](#) untuk praktik terbaik untuk bekerja dengan basis data relasional pada AWS.

Tahap 4: Mengoperasikan dan mengoptimalkan

Ketika database Anda di AWS, Anda harus mengoperasikannya di awan. Anda harus memastikan bahwa Anda mengikuti praktik terbaik untuk bidang-bidang seperti pemantauan, peringatan, backup, dan ketersediaan tinggi. Overhead operasi database rehosted lebih tinggi dari database yang telah replatformed atau refactored untuk menggunakan dikelola AWS layanan database:

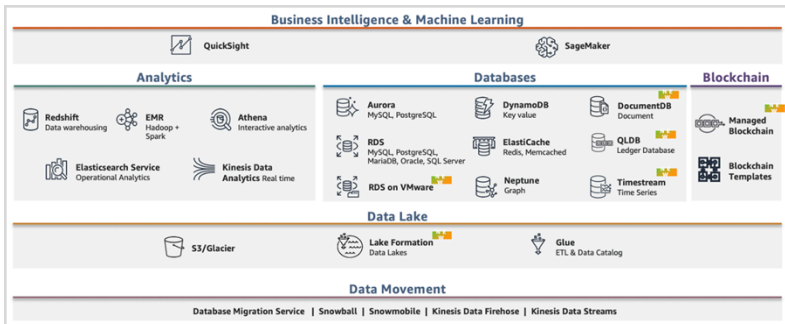
- Basis data yang dihosting ulang berjalan pada instans EC2. Anda bertanggung jawab atas semua tugas manajemen database seperti menyiapkan cadangan, ketersediaan tinggi, dan solusi pemulihan bencana.
- Jika Anda melakukan replatform atau refactor database Anda di Amazon RDS, tugas manajemen database ini hanya memerlukan beberapa klik untuk disiapkan. Ini berarti bahwa administrator database akan menghabiskan lebih sedikit waktu untuk mengelola database di Amazon RDS, dibandingkan dengan mengelola database yang di-host ulang pada instans EC2. Amazon RDS juga menyediakan alat pemantauan kinerja yang disebut Amazon RDS Performance Insights, yang memungkinkan bahkan non-ahli untuk mendeteksi masalah kinerja dengan menggunakan dasbor yang mudah dipahami yang memvisualisasikan beban database.

Apa pun opsi migrasi yang Anda pilih, Amazon CloudWatch memainkan peran yang sangat penting dalam mengumpulkan metrik utama seperti CPU, memori, dan pemanfaatan I/O. Ini juga menyediakan kemampuan untuk mengatur ambang pada metrik dan untuk memulai tindakan ketika ambang batas yang diberikan disilangkan. Misalnya, Anda dapat membuat alarm pada metrik klaster Aurora PostgreSQL, mengatur notifikasi, dan mengambil tindakan untuk mendeteksi dan mematikan instans pembaca yang tidak terpakai atau kurang dimanfaatkan. Menetapkan alarm real-time pada metrik dan peristiwa memungkinkan Anda meminimalkan waktu henti dan potensi dampak bisnis.

Dalam fase mengoperasikan dan mengoptimalkan, Anda dapat memaksimalkan manfaat yang diperoleh dari aplikasi hosting AWS. Aktivitas pengoptimalan dapat mengatasi masalah biaya, kinerja, keamanan, atau ketahanan untuk tumpukan aplikasi Anda. Misalnya, Anda dapat menggunakan fitur penskalaan otomatis untuk menambahkan lebih banyak replika baca selama jam sibuk, dan menghapusnya selama jam sibuk untuk menurunkan biaya. Anda juga dapat menggunakan sejumlah AWS layanan yang terintegrasi secara mulus dengan database Amazon RDS. Misalnya, Anda dapat dengan mudah mengarahkan log mesin basis data ke Amazon CloudWatch Logs untuk analisis.

Setelah Anda berada di AWS Cloud, Anda dapat mulai mengoptimalkan aplikasi Anda dengan memanfaatkan sejumlah besar layanan dan fitur yang dapat Anda putar dengan beberapa klik.

Anda dapat berinovasi lebih cepat, karena Anda dapat memfokuskan sumber daya TI Anda yang sangat berharga dalam mengembangkan aplikasi yang membedakan bisnis Anda dan mengubah pengalaman pengguna Anda, alih-alih berfokus pada pengangkatan infrastruktur dan pusat data yang tidak berdiferensiasi. Diagram berikut menunjukkan beberapa pilihan yang disediakan oleh AWS layanan.



Selain itu, Anda memiliki kemampuan untuk menyebarkan secara global dalam hitungan menit. Misalnya, dengan beberapa klik Anda dapat membuat [Database Global Amazon Aurora](#) yang memungkinkan Anda dengan mudah skala operasi membaca database di seluruh dunia dan menempatkan aplikasi Anda dekat dengan pengguna Anda.

Demikian pula, Anda dapat menggunakan integrasi untuk mendapatkan nilai lebih dari data Anda. Misalnya, Anda dapat menggunakan [kemampuan pembelajaran mesin \(ML-mu\) dalam aplikasi basis data Aurora Anda](#) dengan beberapa langkah sederhana.

Menggunakan AWS Mitra

Migrasi database dapat menjadi proyek yang menantang yang membutuhkan keahlian dan alat. Anda dapat mempercepat migrasi dan waktu Anda untuk menghasilkan melalui kemitraan. [AWS Database Migration Service Mitra](#) memiliki keahlian yang diperlukan untuk membantu pelanggan bermigrasi ke cloud dengan mudah dan aman. Mitra ini memiliki keahlian untuk migrasi homogen seperti Oracle ke Oracle, dan migrasi heterogen antara platform database yang berbeda, seperti Oracle ke Amazon Aurora atau Microsoft SQL Server ke MySQL.

Berdasarkan persyaratan dan preferensi Anda, Anda dapat menggunakan mitra untuk menangani migrasi lengkap atau untuk membantu hanya beberapa aspek migrasi. Selain itu, Anda dapat menggunakan alat dan solusi yang disediakan oleh AWS Mitra untuk membantu migrasi.

Langkah selanjutnya

Untuk informasi selengkapnya tentang memigrasikan beban kerja Oracle Database dan SQL Server Anda, lihat panduan berikut ini di AWS Situs web Prescriptive Guidance:

- [Migrasi database Oracle ke AWS Cloud](#)
- [Migrasi database SQL Server ke AWS Cloud](#)

Untuk petunjuk langkah demi langkah untuk memigrasikan database relasional tertentu, lihat [pola migrasi database](#). Anda dapat menggunakan filter pada halaman tersebut untuk melihat pola oleh AWS layanan (misalnya, migrasi ke Aurora), dengan beban kerja (misalnya, migrasi database Oracle), dengan penggunaan yang direncanakan (produksi atau pilot), atau dengan strategi migrasi (re-architect, rehost, relocate, atau replatform).

Sumber daya

- [Migrasi database Oracle keAWSCloud](#)
- [Memigrasi basis data SQL Server keAWSCloud](#)
- [AWS DMSdokumentasi](#)
- [AWS SCTdokumentasi](#)
- [Buku pedoman migrasi](#)
- [AWSOps basis data](#)
- Informasi umum tentangAWSlayanan basis data terkelola:
 - [Amazon RDS](#)
 - [Amazon Aurora](#)
 - [Amazon RDS for MySQL](#)
 - [Amazon RDS for Oracle](#)
 - [Amazon RDS for PostgreSQL](#)
 - [Amazon RDS for SQL Server](#)
- [Dokumentasi Amazon RDS](#)

Riwayat dokumen

Tabel berikut menjelaskan perubahan signifikan pada dokumen ini. Jika Anda ingin diberi tahu tentang pembaruan di future, Anda dapat berlangganan ke [Umpan RSS](#).

Perubahan	Deskripsi	Tanggal
Diperbarui AWS Informasi WQF	Kami memperbarui Bagian Beban Kerja Kualifikasi dengan informasi terbaru tentang AW SWQF.	5 November 2020
Publikasi awal	—	15 Desember 2019

AWSGlosarium Panduan Preskriptif

Berikut ini adalah istilah yang umum digunakan dalam strategi, panduan, dan pola yang disediakan oleh Panduan AWS Preskriptif. Untuk menyarankan entri, silakan gunakan tautan Berikan umpan balik di akhir glosarium.

Nomor

7 Rs

Tujuh strategi migrasi umum untuk memindahkan aplikasi ke cloud. Strategi ini dibangun di atas 5 Rs yang diidentifikasi Gartner pada tahun 2011 dan terdiri dari yang berikut:

- Refactor/Re-Architect — Memindahkan aplikasi dan memodifikasi arsitekturnya dengan memanfaatkan sepenuhnya fitur cloud-native untuk meningkatkan kelincahan, kinerja, dan skalabilitas. Ini biasanya melibatkan porting sistem operasi dan database. Contoh: Migrasikan database Oracle lokal Anda ke Amazon Aurora PostgreSQL Compatible Edition.
- Replatform (angkat dan bentuk ulang) — Pindahkan aplikasi ke cloud, dan perkenalkan beberapa tingkat pengoptimalan untuk memanfaatkan kemampuan cloud. Contoh: Migrasikan database Oracle lokal Anda ke Amazon Relational Database Service (Amazon RDS) untuk Oracle di Cloud. AWS
- Pembelian kembali (drop and shop) - Beralih ke produk yang berbeda, biasanya dengan beralih dari lisensi tradisional ke model SaaS. Contoh: Migrasikan sistem manajemen hubungan pelanggan (CRM) Anda ke Salesforce.com.
- Rehost (lift dan shift) — Pindahkan aplikasi ke cloud tanpa membuat perubahan apa pun untuk memanfaatkan kemampuan cloud. Contoh: Migrasikan database Oracle lokal Anda ke Oracle pada instans EC2 di Cloud. AWS
- Relokasi (hypervisor-level lift and shift) — Pindahkan infrastruktur ke cloud tanpa membeli perangkat keras baru, menulis ulang aplikasi, atau memodifikasi operasi yang ada. Skenario migrasi ini khusus untuk VMware Cloud onAWS, yang mendukung kompatibilitas mesin virtual (VM) dan portabilitas beban kerja antara lingkungan lokal Anda dan. AWS Anda dapat menggunakan teknologi VMware Cloud Foundation dari pusat data lokal saat memigrasikan infrastruktur ke VMware Cloud. AWS Contoh: Pindahkan hypervisor yang menghosting database Oracle Anda ke VMware Cloud on. AWS
- Pertahankan (kunjungi kembali) - Simpan aplikasi di lingkungan sumber Anda. Ini mungkin termasuk aplikasi yang memerlukan refactoring besar, dan Anda ingin menunda pekerjaan itu

sampai nanti, dan aplikasi lama yang ingin Anda pertahankan, karena tidak ada pembenaran bisnis untuk memigrasikannya.

- Pensiun — Menonaktifkan atau menghapus aplikasi yang tidak lagi diperlukan di lingkungan sumber Anda.

A

ABAC

Lihat [kontrol akses berbasis atribut](#).

layanan abstrak

Lihat [layanan terkelola](#).

ASAM

Lihat [atomisitas, konsistensi, isolasi, daya tahan](#).

migrasi aktif-aktif

Metode migrasi database di mana database sumber dan target tetap sinkron (dengan menggunakan alat replikasi dua arah atau operasi penulisan ganda), dan kedua database menangani transaksi dari menghubungkan aplikasi selama migrasi. Metode ini mendukung migrasi dalam batch kecil yang terkontrol alih-alih memerlukan pemotongan satu kali. Ini lebih fleksibel tetapi membutuhkan lebih banyak pekerjaan daripada migrasi [aktif-pasif](#).

migrasi aktif-pasif

Metode migrasi database di mana database sumber dan target disimpan dalam sinkron, tetapi hanya database sumber yang menangani transaksi dari menghubungkan aplikasi sementara data direplikasi ke database target. Basis data target tidak menerima transaksi apa pun selama migrasi.

fungsi agregat

Fungsi SQL yang beroperasi pada sekelompok baris dan menghitung nilai pengembalian tunggal untuk grup. Contoh fungsi agregat meliputi SUM dan MAX.

AI

Lihat [kecerdasan buatan](#).

AIOps

Lihat [operasi kecerdasan buatan](#).

anonimisasi

Proses menghapus informasi pribadi secara permanen dalam kumpulan data. Anonimisasi dapat membantu melindungi privasi pribadi. Data anonim tidak lagi dianggap sebagai data pribadi.

anti-pola

Solusi yang sering digunakan untuk masalah berulang di mana solusinya kontra-produktif, tidak efektif, atau kurang efektif daripada alternatif.

kontrol aplikasi

Pendekatan keamanan yang memungkinkan penggunaan hanya aplikasi yang disetujui untuk membantu melindungi sistem dari malware.

portofolio aplikasi

Kumpulan informasi rinci tentang setiap aplikasi yang digunakan oleh organisasi, termasuk biaya untuk membangun dan memelihara aplikasi, dan nilai bisnisnya. Informasi ini adalah kunci untuk [penemuan portofolio dan proses analisis dan](#) membantu mengidentifikasi dan memprioritaskan aplikasi yang akan dimigrasi, dimodernisasi, dan dioptimalkan.

kecerdasan buatan (AI)

Bidang ilmu komputer yang didedikasikan untuk menggunakan teknologi komputasi untuk melakukan fungsi kognitif yang biasanya terkait dengan manusia, seperti belajar, memecahkan masalah, dan mengenali pola. Untuk informasi lebih lanjut, lihat [Apa itu Kecerdasan Buatan?](#)

operasi kecerdasan buatan (AIOps)

Proses menggunakan teknik pembelajaran mesin untuk memecahkan masalah operasional, mengurangi insiden operasional dan intervensi manusia, dan meningkatkan kualitas layanan. Untuk informasi selengkapnya tentang cara AIOps digunakan dalam strategi AWS migrasi, lihat [panduan integrasi operasi](#).

enkripsi asimetris

Algoritma enkripsi yang menggunakan sepasang kunci, kunci publik untuk enkripsi dan kunci pribadi untuk dekripsi. Anda dapat berbagi kunci publik karena tidak digunakan untuk dekripsi, tetapi akses ke kunci pribadi harus sangat dibatasi.

atomisitas, konsistensi, isolasi, daya tahan (ACID)

Satu set properti perangkat lunak yang menjamin validitas data dan keandalan operasional database, bahkan dalam kasus kesalahan, kegagalan daya, atau masalah lainnya.

kontrol akses berbasis atribut (ABAC)

Praktik membuat izin berbutir halus berdasarkan atribut pengguna, seperti departemen, peran pekerjaan, dan nama tim. Untuk informasi selengkapnya, lihat [ABAC untuk AWS](#) dokumentasi AWS Identity and Access Management (IAM).

sumber data otoritatif

Lokasi di mana Anda menyimpan versi utama data, yang dianggap sebagai sumber informasi yang paling dapat diandalkan. Anda dapat menyalin data dari sumber data otoritatif ke lokasi lain untuk tujuan memproses atau memodifikasi data, seperti menganonimkan, menyunting, atau membuat nama samaran.

Zona Ketersediaan

Lokasi berbeda di dalam Wilayah AWS yang terisolasi dari kegagalan di Availability Zone lainnya dan menyediakan konektivitas jaringan latensi rendah yang murah ke Availability Zone lainnya di Wilayah yang sama.

AWS Kerangka Adopsi Cloud (AWS CAF)

Kerangka pedoman dan praktik terbaik AWS untuk membantu organisasi mengembangkan rencana yang efisien dan efektif untuk bergerak dengan sukses ke cloud. AWS CAF mengatur panduan ke dalam enam area fokus yang disebut perspektif: bisnis, orang, tata kelola, platform, keamanan, dan operasi. Perspektif bisnis, orang, dan tata kelola fokus pada keterampilan dan proses bisnis; perspektif platform, keamanan, dan operasi fokus pada keterampilan dan proses teknis. Misalnya, perspektif masyarakat menargetkan pemangku kepentingan yang menangani sumber daya manusia (SDM), fungsi kepegawaian, dan manajemen orang. Untuk perspektif ini, AWS CAF memberikan panduan untuk pengembangan, pelatihan, dan komunikasi orang untuk membantu mempersiapkan organisasi untuk adopsi cloud yang sukses. Untuk informasi lebih lanjut, lihat [situs web AWS CAF](#) dan [whitepaper AWS CAF](#).

AWS Kerangka Kualifikasi Beban Kerja (AWS WQF)

Alat yang mengevaluasi beban kerja migrasi database, merekomendasikan strategi migrasi, dan memberikan perkiraan kerja. AWS WQF disertakan dengan AWS Schema Conversion Tool (AWS SCT). Ini menganalisis skema database dan objek kode, kode aplikasi, dependensi, dan karakteristik kinerja, dan memberikan laporan penilaian.

B

BCP

Lihat [perencanaan kontinuitas bisnis](#).

grafik perilaku

Pandangan interaktif yang terpadu tentang perilaku dan interaksi sumber daya dari waktu ke waktu. Anda dapat menggunakan grafik perilaku dengan Amazon Detective untuk memeriksa upaya logon yang gagal, panggilan API yang mencurigakan, dan tindakan serupa. Untuk informasi selengkapnya, lihat [Data dalam grafik perilaku](#) di dokumentasi Detektif.

sistem big-endian

Sistem yang menyimpan byte paling signifikan terlebih dahulu. Lihat juga [endianness](#).

klasifikasi biner

Sebuah proses yang memprediksi hasil biner (salah satu dari dua kelas yang mungkin). Misalnya, model ML Anda mungkin perlu memprediksi masalah seperti “Apakah email ini spam atau bukan spam?” atau “Apakah produk ini buku atau mobil?”

filter mekar

Struktur data probabilistik dan efisien memori yang digunakan untuk menguji apakah suatu elemen adalah anggota dari suatu himpunan.

cabang

Area berisi repositori kode. Cabang pertama yang dibuat dalam repositori adalah cabang utama. Anda dapat membuat cabang baru dari cabang yang ada, dan Anda kemudian dapat mengembangkan fitur atau memperbaiki bug di cabang baru. Cabang yang Anda buat untuk membangun fitur biasanya disebut sebagai cabang fitur. Saat fitur siap dirilis, Anda menggabungkan cabang fitur kembali ke cabang utama. Untuk informasi selengkapnya, lihat [Tentang cabang](#) (GitHub dokumentasi).

akses break-glass

Dalam keadaan luar biasa dan melalui proses yang disetujui, cara cepat bagi pengguna untuk mendapatkan akses ke Akun AWS yang biasanya tidak memiliki izin untuk mengaksesnya. Untuk informasi lebih lanjut, lihat indikator [Implementasikan prosedur break-glass](#) dalam panduan Well-ArchitectedAWS.

strategi brownfield

Infrastruktur yang ada di lingkungan Anda. Saat mengadopsi strategi brownfield untuk arsitektur sistem, Anda merancang arsitektur di sekitar kendala sistem dan infrastruktur saat ini. Jika Anda memperluas infrastruktur yang ada, Anda dapat memadukan strategi brownfield dan [greenfield](#).

cache penyangga

Area memori tempat data yang paling sering diakses disimpan.

kemampuan bisnis

Apa yang dilakukan bisnis untuk menghasilkan nilai (misalnya, penjualan, layanan pelanggan, atau pemasaran). Arsitektur layanan mikro dan keputusan pengembangan dapat didorong oleh kemampuan bisnis. Untuk informasi selengkapnya, lihat bagian [Terorganisir di sekitar kemampuan bisnis](#) dari [Menjalankan layanan mikro kontainer](#) di whitepaper. AWS

perencanaan kelangsungan bisnis (BCP)

Rencana yang membahas dampak potensial dari peristiwa yang mengganggu, seperti migrasi skala besar, pada operasi dan memungkinkan bisnis untuk melanjutkan operasi dengan cepat.

C

KAFE

Lihat [Kerangka Adopsi AWS Cloud](#).

CCoE

Lihat [Cloud Center of Excellence](#).

CDC

Lihat [mengubah pengambilan data](#).

ubah pengambilan data (CDC)

Proses melacak perubahan ke sumber data, seperti tabel database, dan merekam metadata tentang perubahan tersebut. Anda dapat menggunakan CDC untuk berbagai tujuan, seperti mengaudit atau mereplikasi perubahan dalam sistem target untuk mempertahankan sinkronisasi.

rekayasa kekacauan

Sengaja memperkenalkan kegagalan atau peristiwa yang mengganggu untuk menguji ketahanan sistem. Anda dapat menggunakan [AWS Fault Injection Service\(AWS FIS\)](#) untuk melakukan eksperimen yang menekankan AWS beban kerja Anda dan mengevaluasi responsnya.

CI/CD

Lihat [integrasi berkelanjutan dan pengiriman berkelanjutan](#).

klasifikasi

Proses kategorisasi yang membantu menghasilkan prediksi. Model ML untuk masalah klasifikasi memprediksi nilai diskrit. Nilai diskrit selalu berbeda satu sama lain. Misalnya, model mungkin perlu mengevaluasi apakah ada mobil dalam gambar atau tidak.

Enkripsi sisi klien

Enkripsi data secara lokal, sebelum target Layanan AWS menerimanya.

Cloud Center of Excellence (CCoE)

Tim multi-disiplin yang mendorong upaya adopsi cloud di seluruh organisasi, termasuk mengembangkan praktik terbaik cloud, memobilisasi sumber daya, menetapkan jadwal migrasi, dan memimpin organisasi melalui transformasi skala besar. Untuk informasi selengkapnya, lihat [posting CCoE](#) di Blog Strategi AWS Cloud Enterprise.

komputasi cloud

Teknologi cloud yang biasanya digunakan untuk penyimpanan data jarak jauh dan manajemen perangkat IoT. Cloud computing umumnya terhubung ke teknologi [edge computing](#).

model operasi cloud

Dalam organisasi TI, model operasi yang digunakan untuk membangun, mematangkan, dan mengoptimalkan satu atau lebih lingkungan cloud. Untuk informasi selengkapnya, lihat [Membangun Model Operasi Cloud Anda](#).

tahap adopsi cloud

Empat fase yang biasanya dilalui organisasi ketika mereka bermigrasi ke AWS Cloud:

- Proyek — Menjalankan beberapa proyek terkait cloud untuk bukti konsep dan tujuan pembelajaran
- Foundation — Melakukan investasi dasar untuk meningkatkan adopsi cloud Anda (misalnya, membuat landing zone, mendefinisikan CCoE, membuat model operasi)

- Migrasi — Migrasi aplikasi individual
- Re-invention — Mengoptimalkan produk dan layanan, dan berinovasi di cloud

Tahapan ini didefinisikan oleh Stephen Orban dalam posting blog The [Journey Toward Cloud-First & the Stages of Adoption](#) di blog AWS Cloud Enterprise Strategy. Untuk informasi tentang bagaimana kaitannya dengan strategi AWS migrasi, lihat [panduan kesiapan migrasi](#).

CMDB

Lihat [database manajemen konfigurasi](#).

repositori kode

Lokasi di mana kode sumber dan aset lainnya, seperti dokumentasi, sampel, dan skrip, disimpan dan diperbarui melalui proses kontrol versi. Repositori cloud umum termasuk GitHub atau AWS CodeCommit. Setiap versi kode disebut cabang. Dalam struktur layanan mikro, setiap repositori dikhususkan untuk satu bagian fungsionalitas. Pipa CI/CD tunggal dapat menggunakan beberapa repositori.

cache dingin

Cache buffer yang kosong, tidak terisi dengan baik, atau berisi data basi atau tidak relevan. Ini mempengaruhi kinerja karena instance database harus membaca dari memori utama atau disk, yang lebih lambat daripada membaca dari cache buffer.

data dingin

Data yang jarang diakses dan biasanya historis. Saat menanyakan jenis data ini, kueri lambat biasanya dapat diterima. Memindahkan data ini ke tingkat atau kelas penyimpanan yang berkinerja lebih rendah dan lebih murah dapat mengurangi biaya.

visi komputer

Bidang AI yang digunakan oleh mesin untuk mengidentifikasi orang, tempat, dan benda dalam gambar dengan akurasi pada atau di atas tingkat manusia. Sering dibangun dengan model pembelajaran mendalam, ini mengotomatiskan ekstraksi, analisis, klasifikasi, dan pemahaman informasi yang berguna dari satu gambar atau urutan gambar.

database manajemen konfigurasi (CMDB)

Repositori yang menyimpan dan mengelola informasi tentang database dan lingkungan TI, termasuk komponen perangkat keras dan perangkat lunak dan konfigurasinya. Anda biasanya menggunakan data dari CMDB dalam penemuan portofolio dan tahap analisis migrasi.

paket kesesuaian

Kumpulan AWS Config aturan dan tindakan remediasi yang dapat Anda kumpulkan untuk menyesuaikan kepatuhan dan pemeriksaan keamanan Anda. Anda dapat menerapkan paket kesesuaian sebagai entitas tunggal di Akun AWS dan Region, atau di seluruh organisasi, dengan menggunakan templat YAMM. Untuk informasi selengkapnya, lihat [Paket kesesuaian dalam dokumentasi](#). AWS Config

integrasi berkelanjutan dan pengiriman berkelanjutan (CI/CD)

Proses mengotomatiskan sumber, membangun, menguji, pementasan, dan tahap produksi dari proses rilis perangkat lunak. CI/CD umumnya digambarkan sebagai pipa. CI/CD dapat membantu Anda mengotomatiskan proses, meningkatkan produktivitas, meningkatkan kualitas kode, dan memberikan lebih cepat. Untuk informasi lebih lanjut, lihat [Manfaat pengiriman berkelanjutan](#). CD juga dapat berarti penerapan berkelanjutan. Untuk informasi selengkapnya, lihat [Continuous Delivery vs Continuous Deployment](#).

D

data saat istirahat

Data yang stasioner di jaringan Anda, seperti data yang ada di penyimpanan.

klasifikasi data

Proses untuk mengidentifikasi dan mengkategorikan data dalam jaringan Anda berdasarkan kekritisannya dan sensitivitasnya. Ini adalah komponen penting dari setiap strategi manajemen risiko keamanan siber karena membantu Anda menentukan perlindungan dan kontrol retensi yang tepat untuk data. Klasifikasi data adalah komponen pilar keamanan dalam AWS Well-Architected Framework. Untuk informasi selengkapnya, lihat [Klasifikasi data](#).

penyimpangan data

Variasi yang berarti antara data produksi dan data yang digunakan untuk melatih model ML, atau perubahan yang berarti dalam data input dari waktu ke waktu. Penyimpangan data dapat mengurangi kualitas, akurasi, dan keadilan keseluruhan dalam prediksi model ML.

data dalam transit

Data yang aktif bergerak melalui jaringan Anda, seperti antara sumber daya jaringan.

minimalisasi data

Prinsip pengumpulan dan pemrosesan hanya data yang sangat diperlukan. Mempraktikkan minimalisasi data AWS Cloud dapat mengurangi risiko privasi, biaya, dan jejak karbon analitik Anda.

perimeter data

Satu set pagar pembatas pencegahan di AWS lingkungan Anda yang membantu memastikan bahwa hanya identitas tepercaya yang mengakses sumber daya tepercaya dari jaringan yang diharapkan. Untuk informasi selengkapnya, lihat [Membangun perimeter data pada AWS](#).

prapemrosesan data

Untuk mengubah data mentah menjadi format yang mudah diuraikan oleh model ML Anda. Preprocessing data dapat berarti menghapus kolom atau baris tertentu dan menangani nilai yang hilang, tidak konsisten, atau duplikat.

asal data

Proses melacak asal dan riwayat data sepanjang siklus hidupnya, seperti bagaimana data dihasilkan, ditransmisikan, dan disimpan.

subjek data

Individu yang datanya dikumpulkan dan diproses.

gudang data

Sistem manajemen data yang mendukung intelijen bisnis, seperti analitik. Gudang data biasanya berisi sejumlah besar data historis, dan biasanya digunakan untuk kueri dan analisis.

bahasa definisi database (DDL)

Pernyataan atau perintah untuk membuat atau memodifikasi struktur tabel dan objek dalam database.

bahasa manipulasi basis data (DHTML)

Pernyataan atau perintah untuk memodifikasi (memasukkan, memperbarui, dan menghapus) informasi dalam database.

DDL

Lihat [bahasa definisi database](#).

ansambel yang dalam

Untuk menggabungkan beberapa model pembelajaran mendalam untuk prediksi. Anda dapat menggunakan ansambel dalam untuk mendapatkan prediksi yang lebih akurat atau untuk memperkirakan ketidakpastian dalam prediksi.

pembelajaran mendalam

Subbidang ML yang menggunakan beberapa lapisan jaringan saraf tiruan untuk mengidentifikasi pemetaan antara data input dan variabel target yang diinginkan.

defense-in-depth

Pendekatan keamanan informasi di mana serangkaian mekanisme dan kontrol keamanan dilapisi dengan cermat di seluruh jaringan komputer untuk melindungi kerahasiaan, integritas, dan ketersediaan jaringan dan data di dalamnya. Saat Anda mengadopsi strategi ini AWS, Anda menambahkan beberapa kontrol pada lapisan AWS Organizations struktur yang berbeda untuk membantu mengamankan sumber daya. Misalnya, defense-in-depth pendekatan mungkin menggabungkan otentikasi multi-faktor, segmentasi jaringan, dan enkripsi.

administrator yang didelegasikan

Di AWS Organizations, layanan yang kompatibel dapat mendaftarkan akun AWS anggota untuk mengelola akun organisasi dan mengelola izin untuk layanan tersebut. Akun ini disebut administrator yang didelegasikan untuk layanan itu. Untuk informasi selengkapnya dan daftar layanan yang kompatibel, lihat [Layanan yang berfungsi dengan AWS Organizations](#) AWS Organizations dokumentasi.

deployment

Proses pembuatan aplikasi, fitur baru, atau perbaikan kode tersedia di lingkungan target. Deployment melibatkan penerapan perubahan dalam basis kode dan kemudian membangun dan menjalankan basis kode itu di lingkungan aplikasi.

lingkungan pengembangan

Lihat [lingkungan](#).

kontrol detektif

Kontrol keamanan yang dirancang untuk mendeteksi, mencatat, dan memperingatkan setelah suatu peristiwa terjadi. Kontrol ini adalah garis pertahanan kedua, memperingatkan Anda tentang peristiwa keamanan yang melewati kontrol pencegahan yang ada. Untuk informasi selengkapnya, lihat Kontrol [Detektif dalam Menerapkan kontrol](#) keamanan pada. AWS

pemetaan aliran nilai pengembangan (DVSM)

Proses yang digunakan untuk mengidentifikasi dan memprioritaskan kendala yang berdampak buruk pada kecepatan dan kualitas dalam siklus hidup pengembangan perangkat lunak. DVSM memperluas proses pemetaan aliran nilai yang awalnya dirancang untuk praktik manufaktur ramping. Ini berfokus pada langkah-langkah dan tim yang diperlukan untuk menciptakan dan memindahkan nilai melalui proses pengembangan perangkat lunak.

kembar digital

Representasi virtual dari sistem dunia nyata, seperti bangunan, pabrik, peralatan industri, atau jalur produksi. Kembar digital mendukung pemeliharaan prediktif, pemantauan jarak jauh, dan optimalisasi produksi.

tabel dimensi

Dalam [skema bintang](#), tabel yang lebih kecil yang berisi atribut data tentang data kuantitatif dalam tabel fakta. Atribut tabel dimensi biasanya bidang teks atau angka diskrit yang berperilaku seperti teks. Atribut ini biasanya digunakan untuk pembatasan kueri, pemfilteran, dan pelabelan set hasil.

musibah

Peristiwa yang mencegah beban kerja atau sistem memenuhi tujuan bisnisnya di lokasi utama yang digunakan. Peristiwa ini dapat berupa bencana alam, kegagalan teknis, atau akibat dari tindakan manusia, seperti kesalahan konfigurasi yang tidak disengaja atau serangan malware.

pemulihan bencana (DR)

Strategi dan proses yang Anda gunakan untuk meminimalkan downtime dan kehilangan data yang disebabkan oleh [bencana](#). Untuk informasi selengkapnya, lihat [Disaster Recovery of Workloads on AWS: Recovery in the Cloud in the AWS Well-Architected Framework](#).

DML~

Lihat [bahasa manipulasi database](#).

desain berbasis domain

Pendekatan untuk mengembangkan sistem perangkat lunak yang kompleks dengan menghubungkan komponennya ke domain yang berkembang, atau tujuan bisnis inti, yang dilayani oleh setiap komponen. Konsep ini diperkenalkan oleh Eric Evans dalam bukunya, *Domain-Driven Design: Tackling Complexity in the Heart of Software* (Boston: Addison-Wesley Professional, 2003). Untuk informasi tentang cara menggunakan desain berbasis domain dengan pola gambar

pengecek, lihat Memodernisasi layanan web [Microsoft ASP.NET \(ASMX\) lama secara bertahap](#) menggunakan container dan Amazon API Gateway.

DR

Lihat [pemulihan bencana](#).

deteksi drift

Melacak penyimpangan dari konfigurasi dasar. Misalnya, Anda dapat menggunakan AWS CloudFormation untuk [mendeteksi penyimpangan dalam sumber daya sistem](#), atau Anda dapat menggunakannya AWS Control Tower untuk [mendeteksi perubahan di landing zone](#) yang mungkin memengaruhi kepatuhan terhadap persyaratan tata kelola.

DVSM

Lihat [pemetaan aliran nilai pengembangan](#).

E

EDA

Lihat [analisis data eksplorasi](#).

komputasi tepi

Teknologi yang meningkatkan daya komputasi untuk perangkat pintar di tepi jaringan IoT. Jika dibandingkan dengan [komputasi awan](#), komputasi tepi dapat mengurangi latensi komunikasi dan meningkatkan waktu respons.

enkripsi

Proses komputasi yang mengubah data plaintext, yang dapat dibaca manusia, menjadi ciphertext.

kunci enkripsi

String kriptografi dari bit acak yang dihasilkan oleh algoritma enkripsi. Panjang kunci dapat bervariasi, dan setiap kunci dirancang agar tidak dapat diprediksi dan unik.

endianness

Urutan byte disimpan dalam memori komputer. Sistem big-endian menyimpan byte paling signifikan terlebih dahulu. Sistem little-endian menyimpan byte paling tidak signifikan terlebih dahulu.

titik akhir

Lihat [titik akhir layanan](#).

layanan endpoint

Layanan yang dapat Anda host di cloud pribadi virtual (VPC) untuk dibagikan dengan pengguna lain. Anda dapat membuat layanan endpoint dengan AWS PrivateLink dan memberikan izin kepada prinsipal lain Akun AWS atau ke AWS Identity and Access Management (IAM). Akun atau prinsipal ini dapat terhubung ke layanan endpoint Anda secara pribadi dengan membuat titik akhir VPC antarmuka. Untuk informasi selengkapnya, lihat [Membuat layanan titik akhir](#) di dokumentasi Amazon Virtual Private Cloud (Amazon VPC).

enkripsi amplop

Proses mengenkripsi kunci enkripsi dengan kunci enkripsi lain. Untuk informasi selengkapnya, lihat [Enkripsi amplop](#) dalam dokumentasi AWS Key Management Service (AWS KMS).

lingkungan

Sebuah contoh dari aplikasi yang sedang berjalan. Berikut ini adalah jenis lingkungan yang umum dalam komputasi awan:

- Development Environment — Sebuah contoh dari aplikasi yang berjalan yang hanya tersedia untuk tim inti yang bertanggung jawab untuk memelihara aplikasi. Lingkungan pengembangan digunakan untuk menguji perubahan sebelum mempromosikannya ke lingkungan atas. Jenis lingkungan ini kadang-kadang disebut sebagai lingkungan pengujian.
- lingkungan yang lebih rendah — Semua lingkungan pengembangan untuk aplikasi, seperti yang digunakan untuk build awal dan pengujian.
- lingkungan produksi — Sebuah contoh dari aplikasi yang berjalan yang pengguna akhir dapat mengakses. Dalam pipa CI/CD, lingkungan produksi adalah lingkungan penyebaran terakhir.
- lingkungan atas — Semua lingkungan yang dapat diakses oleh pengguna selain tim pengembangan inti. Ini dapat mencakup lingkungan produksi, lingkungan praproduksi, dan lingkungan untuk pengujian penerimaan pengguna.

epik

Dalam metodologi tangkas, kategori fungsional yang membantu mengatur dan memprioritaskan pekerjaan Anda. Epik memberikan deskripsi tingkat tinggi tentang persyaratan dan tugas implementasi. Misalnya, epos keamanan AWS CAF mencakup manajemen identitas dan akses, kontrol detektif, keamanan infrastruktur, perlindungan data, dan respons insiden. Untuk informasi selengkapnya tentang epos dalam strategi AWS migrasi, lihat [panduan implementasi program](#).

analisis data eksplorasi (EDA)

Proses menganalisis dataset untuk memahami karakteristik utamanya. Anda mengumpulkan atau mengumpulkan data dan kemudian melakukan penyelidikan awal untuk menemukan pola, mendeteksi anomali, dan memeriksa asumsi. EDA dilakukan dengan menghitung statistik ringkasan dan membuat visualisasi data.

F

tabel fakta

Tabel tengah dalam [skema bintang](#). Ini menyimpan data kuantitatif tentang operasi bisnis. Biasanya, tabel fakta berisi dua jenis kolom: kolom yang berisi ukuran dan yang berisi kunci asing ke tabel dimensi.

gagal cepat

Filosofi yang menggunakan pengujian yang sering dan bertahap untuk mengurangi siklus hidup pengembangan. Ini adalah bagian penting dari pendekatan tangkas.

batas isolasi kesalahan

Dalam AWS Cloud, batas seperti Availability Zone, Wilayah AWS, control plane, atau data plane yang membatasi efek kegagalan dan membantu meningkatkan ketahanan beban kerja. Untuk informasi selengkapnya, lihat [Batas Isolasi AWS Kesalahan](#).

cabang fitur

Lihat [cabang](#).

fitur

Data input yang Anda gunakan untuk membuat prediksi. Misalnya, dalam konteks manufaktur, fitur bisa berupa gambar yang diambil secara berkala dari lini manufaktur.

pentingnya fitur

Seberapa signifikan fitur untuk prediksi model. Ini biasanya dinyatakan sebagai skor numerik yang dapat dihitung melalui berbagai teknik, seperti Shapley Additive Explanations (SHAP) dan gradien terintegrasi. Untuk informasi lebih lanjut, lihat [Interpretabilitas model pembelajaran mesin dengan AWS](#).

transformasi fitur

Untuk mengoptimalkan data untuk proses ML, termasuk memperkaya data dengan sumber tambahan, menskalakan nilai, atau mengekstrak beberapa set informasi dari satu bidang data. Hal ini memungkinkan model ML untuk mendapatkan keuntungan dari data. Misalnya, jika Anda memecah tanggal “2021-05-27 00:15:37” menjadi “2021”, “Mei”, “Kamis”, dan “15”, Anda dapat membantu algoritme pembelajaran mempelajari pola bernuansa yang terkait dengan komponen data yang berbeda.

FGAC

Lihat kontrol [akses berbutir halus](#).

kontrol akses berbutir halus (FGAC)

Penggunaan beberapa kondisi untuk mengizinkan atau menolak permintaan akses.

migrasi flash-cut

Metode migrasi database yang menggunakan replikasi data berkelanjutan melalui [pengambilan data perubahan](#) untuk memigrasikan data dalam waktu sesingkat mungkin, alih-alih menggunakan pendekatan bertahap. Tujuannya adalah untuk menjaga downtime seminimal mungkin.

G

pemblokiran geografis

Lihat [pembatasan geografis](#).

pembatasan geografis (pemblokiran geografis)

Di Amazon CloudFront, opsi untuk mencegah pengguna di negara tertentu mengakses distribusi konten. Anda dapat menggunakan daftar izinkan atau daftar blokir untuk menentukan negara yang disetujui dan dilarang. Untuk informasi selengkapnya, lihat [Membatasi distribusi geografis konten Anda](#) dalam dokumentasi. CloudFront

Alur kerja Gitflow

Pendekatan di mana lingkungan bawah dan atas menggunakan cabang yang berbeda dalam repositori kode sumber. Alur kerja Gitflow dianggap warisan, dan [alur kerja berbasis batang](#) adalah pendekatan modern yang disukai.

strategi greenfield

Tidak adanya infrastruktur yang ada di lingkungan baru. [Saat mengadopsi strategi greenfield untuk arsitektur sistem, Anda dapat memilih semua teknologi baru tanpa batasan kompatibilitas dengan infrastruktur yang ada, juga dikenal sebagai brownfield.](#) Jika Anda memperluas infrastruktur yang ada, Anda dapat memadukan strategi brownfield dan greenfield.

pagar pembatas

Aturan tingkat tinggi yang membantu mengatur sumber daya, kebijakan, dan kepatuhan di seluruh unit organisasi (OU). Pagar pembatas preventif menegakkan kebijakan untuk memastikan keselarasan dengan standar kepatuhan. Mereka diimplementasikan dengan menggunakan kebijakan kontrol layanan dan batas izin IAM. Detective guardrails mendeteksi pelanggaran kebijakan dan masalah kepatuhan, dan menghasilkan peringatan untuk remediasi. Mereka diimplementasikan dengan menggunakan AWS Config, AWS Security Hub, Amazon GuardDuty, AWS Trusted Advisor, Amazon Inspector, dan pemeriksaan khusus AWS Lambda.

H

HA

Lihat [ketersediaan tinggi](#).

migrasi database heterogen

Memigrasi database sumber Anda ke database target yang menggunakan mesin database yang berbeda (misalnya, Oracle ke Amazon Aurora). Migrasi heterogen biasanya merupakan bagian dari upaya arsitektur ulang, dan mengubah skema dapat menjadi tugas yang kompleks. [AWS menyediakan AWS SCT](#) yang membantu dengan konversi skema.

ketersediaan tinggi (HA)

Kemampuan beban kerja untuk beroperasi terus menerus, tanpa intervensi, jika terjadi tantangan atau bencana. Sistem HA dirancang untuk gagal secara otomatis, secara konsisten memberikan kinerja berkualitas tinggi, dan menangani beban dan kegagalan yang berbeda dengan dampak kinerja minimal.

modernisasi sejarawan

Pendekatan yang digunakan untuk memodernisasi dan meningkatkan sistem teknologi operasional (OT) untuk melayani kebutuhan industri manufaktur dengan lebih baik. Sejarawan

adalah jenis database yang digunakan untuk mengumpulkan dan menyimpan data dari berbagai sumber di pabrik.

migrasi database homogen

Memigrasi database sumber Anda ke database target yang berbagi mesin database yang sama (misalnya, Microsoft SQL Server ke Amazon RDS for SQL Server). Migrasi homogen biasanya merupakan bagian dari upaya rehosting atau replatforming. Anda dapat menggunakan utilitas database asli untuk memigrasi skema.

data panas

Data yang sering diakses, seperti data real-time atau data translasi terbaru. Data ini biasanya memerlukan tingkat atau kelas penyimpanan berkinerja tinggi untuk memberikan respons kueri yang cepat.

perbaikan terbaru

Perbaikan mendesak untuk masalah kritis dalam lingkungan produksi. Karena urgensinya, perbaikan terbaru biasanya dibuat di luar alur kerja DevOps rilis biasa.

periode hypercare

Segera setelah cutover, periode waktu ketika tim migrasi mengelola dan memantau aplikasi yang dimigrasi di cloud untuk mengatasi masalah apa pun. Biasanya, periode ini panjangnya 1-4 hari. Pada akhir periode hypercare, tim migrasi biasanya mentransfer tanggung jawab untuk aplikasi ke tim operasi cloud.

|

IAC

Lihat [infrastruktur sebagai kode](#).

kebijakan berbasis identitas

Kebijakan yang dilampirkan pada satu atau beberapa prinsip IAM yang mendefinisikan izin mereka dalam lingkungan. AWS Cloud

aplikasi idle

Aplikasi yang memiliki penggunaan CPU dan memori rata-rata antara 5 dan 20 persen selama periode 90 hari. Dalam proyek migrasi, adalah umum untuk menghentikan aplikasi ini atau mempertahankannya di tempat.

|

IloT

Lihat [Internet of Things industri](#).

infrastruktur yang tidak dapat diubah

Model yang menyebarkan infrastruktur baru untuk beban kerja produksi alih-alih memperbarui, menambal, atau memodifikasi infrastruktur yang ada. [Infrastruktur yang tidak dapat diubah secara inheren lebih konsisten, andal, dan dapat diprediksi daripada infrastruktur yang dapat berubah](#). Untuk informasi selengkapnya, lihat praktik terbaik [Deploy using immutable infrastructure](#) di AWS Well-Architected Framework.

masuk (masuknya) VPC

Dalam arsitektur AWS multi-akun, VPC yang menerima, memeriksa, dan merutekan koneksi jaringan dari luar aplikasi. [Arsitektur Referensi AWS Keamanan](#) merekomendasikan pengaturan akun Jaringan Anda dengan VPC masuk, keluar, dan inspeksi untuk melindungi antarmuka dua arah antara aplikasi Anda dan internet yang lebih luas.

migrasi inkremental

Strategi cutover di mana Anda memigrasikan aplikasi Anda dalam bagian-bagian kecil alih-alih melakukan satu cutover penuh. Misalnya, Anda mungkin hanya memindahkan beberapa layanan mikro atau pengguna ke sistem baru pada awalnya. Setelah Anda memverifikasi bahwa semuanya berfungsi dengan baik, Anda dapat secara bertahap memindahkan layanan mikro atau pengguna tambahan hingga Anda dapat menonaktifkan sistem lama Anda. Strategi ini mengurangi risiko yang terkait dengan migrasi besar.

infrastruktur

Semua sumber daya dan aset yang terkandung dalam lingkungan aplikasi.

infrastruktur sebagai kode (IAC)

Proses penyediaan dan pengelolaan infrastruktur aplikasi melalui satu set file konfigurasi. IAC dirancang untuk membantu Anda memusatkan manajemen infrastruktur, menstandarisasi sumber daya, dan menskalakan dengan cepat sehingga lingkungan baru dapat diulang, andal, dan konsisten.

Internet of Things industri (IIoT)

Penggunaan sensor dan perangkat yang terhubung ke internet di sektor industri, seperti manufaktur, energi, otomotif, perawatan kesehatan, ilmu kehidupan, dan pertanian. Untuk

informasi selengkapnya, lihat [Membangun strategi transformasi digital Internet of Things \(IIoT\) industri](#).

inspeksi VPC

Dalam arsitektur AWS multi-akun, VPC terpusat yang mengelola inspeksi lalu lintas jaringan antara VPC (dalam hal yang sama atau berbedaWilayah AWS), internet, dan jaringan lokal. [Arsitektur Referensi AWS Keamanan](#) merekomendasikan pengaturan akun Jaringan Anda dengan VPC masuk, keluar, dan inspeksi untuk melindungi antarmuka dua arah antara aplikasi Anda dan internet yang lebih luas.

Internet of Things (IoT)

Jaringan objek fisik yang terhubung dengan sensor atau prosesor tertanam yang berkomunikasi dengan perangkat dan sistem lain melalui internet atau melalui jaringan komunikasi lokal. Untuk informasi lebih lanjut, lihat [Apa itu IoT?](#)

interpretabilitas

Karakteristik model pembelajaran mesin yang menggambarkan sejauh mana manusia dapat memahami bagaimana prediksi model bergantung pada inputnya. Untuk informasi selengkapnya, lihat [Interpretabilitas model pembelajaran mesin dengan AWS](#).

IoT

Lihat [Internet of Things](#).

Perpustakaan informasi TI (ITIL)

Serangkaian praktik terbaik untuk memberikan layanan TI dan menyelaraskan layanan ini dengan persyaratan bisnis. ITIL menyediakan dasar untuk ITSM.

Manajemen layanan TI (ITSM)

Kegiatan yang terkait dengan merancang, menerapkan, mengelola, dan mendukung layanan TI untuk suatu organisasi. Untuk informasi tentang mengintegrasikan operasi cloud dengan alat ITSM, lihat panduan [integrasi operasi](#).

ITIL

Lihat [perpustakaan informasi TI](#).

ITSM

Lihat [manajemen layanan TI](#).

L

kontrol akses berbasis label (LBAC)

Implementasi kontrol akses wajib (MAC) di mana pengguna dan data itu sendiri masing-masing secara eksplisit diberi nilai label keamanan. Persimpangan antara label keamanan pengguna dan label keamanan data menentukan baris dan kolom mana yang dapat dilihat oleh pengguna.

landing zone

Landing zone adalah AWS lingkungan multi-akun yang dirancang dengan baik yang dapat diskalakan dan aman. Ini adalah titik awal dari mana organisasi Anda dapat dengan cepat meluncurkan dan menyebarkan beban kerja dan aplikasi dengan percaya diri dalam lingkungan keamanan dan infrastruktur mereka. Untuk informasi selengkapnya tentang zona pendaratan, lihat [Menyiapkan lingkungan multi-akun AWS yang aman dan dapat diskalakan](#).

migrasi besar

Migrasi 300 atau lebih server.

LBAC

Lihat [kontrol akses berbasis label](#).

hak istimewa paling sedikit

Praktik keamanan terbaik untuk memberikan izin minimum yang diperlukan untuk melakukan tugas. Untuk informasi selengkapnya, lihat [Menerapkan izin hak istimewa terkecil dalam dokumentasi IAM](#).

angkat dan geser

Lihat [7 Rs](#).

sistem endian kecil

Sistem yang menyimpan byte paling tidak signifikan terlebih dahulu. Lihat juga [endianness](#).

lingkungan yang lebih rendah

Lihat [lingkungan](#).

M

pembelajaran mesin (ML)

Jenis kecerdasan buatan yang menggunakan algoritma dan teknik untuk pengenalan pola dan pembelajaran. ML menganalisis dan belajar dari data yang direkam, seperti data Internet of Things (IoT), untuk menghasilkan model statistik berdasarkan pola. Untuk informasi selengkapnya, lihat [Machine Learning](#).

cabang utama

Lihat [cabang](#).

layanan terkelola

Layanan AWS yang AWS mengoperasikan lapisan infrastruktur, sistem operasi, dan platform, dan Anda mengakses titik akhir untuk menyimpan dan mengambil data. Amazon Simple Storage Service (Amazon S3) dan Amazon DynamoDB adalah contoh layanan terkelola. Ini juga dikenal sebagai layanan abstrak.

PETA

Lihat [Program Percepatan Migrasi](#).

mekanisme

Proses lengkap di mana Anda membuat alat, mendorong adopsi alat, dan kemudian memeriksa hasilnya untuk melakukan penyesuaian. Mekanisme adalah siklus yang memperkuat dan meningkatkan dirinya sendiri saat beroperasi. Untuk informasi lebih lanjut, lihat [Membangun mekanisme](#) di AWS Well-Architected Framework.

akun anggota

Semua Akun AWS selain akun manajemen yang merupakan bagian dari organisasi di AWS Organizations. Akun dapat menjadi anggota dari hanya satu organisasi pada suatu waktu.

layanan mikro

Layanan kecil dan independen yang berkomunikasi melalui API yang terdefinisi dengan baik dan biasanya dimiliki oleh tim kecil yang mandiri. Misalnya, sistem asuransi mungkin mencakup layanan mikro yang memetakan kemampuan bisnis, seperti penjualan atau pemasaran, atau subdomain, seperti pembelian, klaim, atau analitik. Manfaat layanan mikro termasuk kelincahan, penskalaan yang fleksibel, penyebaran yang mudah, kode yang dapat digunakan kembali,

dan ketahanan. Untuk informasi selengkapnya, lihat [Mengintegrasikan layanan mikro dengan menggunakan layanan tanpa AWS server](#).

arsitektur microservices

Pendekatan untuk membangun aplikasi dengan komponen independen yang menjalankan setiap proses aplikasi sebagai layanan mikro. Layanan mikro ini berkomunikasi melalui antarmuka yang terdefinisi dengan baik dengan menggunakan API ringan. Setiap layanan mikro dalam arsitektur ini dapat diperbarui, digunakan, dan diskalakan untuk memenuhi permintaan fungsi tertentu dari suatu aplikasi. Untuk informasi selengkapnya, lihat [Menerapkan layanan mikro di AWS](#).

Program Percepatan Migrasi (MAP)

AWS Program yang menyediakan dukungan konsultasi, pelatihan, dan layanan untuk membantu organisasi membangun fondasi operasional yang kuat untuk pindah ke cloud, dan untuk membantu mengimbangi biaya awal migrasi. MAP mencakup metodologi migrasi untuk mengeksekusi migrasi lama dengan cara metodis dan seperangkat alat untuk mengotomatisasi dan mempercepat skenario migrasi umum.

migrasi dalam skala

Proses memindahkan sebagian besar portofolio aplikasi ke cloud dalam gelombang, dengan lebih banyak aplikasi bergerak pada tingkat yang lebih cepat di setiap gelombang. Fase ini menggunakan praktik terbaik dan pelajaran yang dipetik dari fase sebelumnya untuk mengimplementasikan pabrik migrasi tim, alat, dan proses untuk merampingkan migrasi beban kerja melalui otomatisasi dan pengiriman tangkas. Ini adalah fase ketiga dari [strategi AWS migrasi](#).

pabrik migrasi

Tim lintas fungsi yang merampingkan migrasi beban kerja melalui pendekatan otomatis dan gesit. Tim pabrik migrasi biasanya mencakup operasi, analis dan pemilik bisnis, insinyur migrasi, pengembang, dan DevOps profesional yang bekerja di sprint. Antara 20 dan 50 persen portofolio aplikasi perusahaan terdiri dari pola berulang yang dapat dioptimalkan dengan pendekatan pabrik. Untuk informasi selengkapnya, lihat [diskusi tentang pabrik migrasi](#) dan [panduan Pabrik Migrasi Cloud](#) di kumpulan konten ini.

metadata migrasi

Informasi tentang aplikasi dan server yang diperlukan untuk menyelesaikan migrasi. Setiap pola migrasi memerlukan satu set metadata migrasi yang berbeda. Contoh metadata migrasi termasuk subnet target, grup keamanan, dan akun. AWS

pola migrasi

Tugas migrasi berulang yang merinci strategi migrasi, tujuan migrasi, dan aplikasi atau layanan migrasi yang digunakan. Contoh: Rehost migrasi ke Amazon EC2 AWS dengan Layanan Migrasi Aplikasi.

Penilaian Portofolio Migrasi (MPA)

Alat online yang menyediakan informasi untuk memvalidasi kasus bisnis untuk bermigrasi ke Cloud. AWS MPA menyediakan penilaian portofolio terperinci (ukuran kanan server, harga, perbandingan TCO, analisis biaya migrasi) serta perencanaan migrasi (analisis data aplikasi dan pengumpulan data, pengelompokan aplikasi, prioritas migrasi, dan perencanaan gelombang). [Alat MPA](#) (memerlukan login) tersedia gratis untuk semua AWS konsultan dan konsultan APN Partner.

Penilaian Kesiapan Migrasi (MRA)

Proses mendapatkan wawasan tentang status kesiapan cloud organisasi, mengidentifikasi kekuatan dan kelemahan, dan membangun rencana aksi untuk menutup kesenjangan yang diidentifikasi, menggunakan CAF. AWS Untuk informasi selengkapnya, lihat [panduan kesiapan migrasi](#). MRA adalah tahap pertama dari [strategi AWS migrasi](#).

strategi migrasi

Pendekatan yang digunakan untuk memigrasikan beban kerja ke Cloud. AWS Untuk informasi lebih lanjut, lihat entri [7 Rs](#) di glosarium ini dan lihat [Memobilisasi organisasi Anda untuk mempercepat](#) migrasi skala besar.

ML

Lihat [pembelajaran mesin](#).

MPA

Lihat [Penilaian Portofolio Migrasi](#).

modernisasi

Mengubah aplikasi usang (warisan atau monolitik) dan infrastrukturnya menjadi sistem yang gesit, elastis, dan sangat tersedia di cloud untuk mengurangi biaya, mendapatkan efisiensi, dan memanfaatkan inovasi. Untuk informasi selengkapnya, lihat [Strategi untuk memodernisasi aplikasi di Cloud. AWS](#)

penilaian kesiapan modernisasi

Evaluasi yang membantu menentukan kesiapan modernisasi aplikasi organisasi; mengidentifikasi manfaat, risiko, dan dependensi; dan menentukan seberapa baik organisasi dapat mendukung

keadaan masa depan aplikasi tersebut. Hasil penilaian adalah cetak biru arsitektur target, peta jalan yang merinci fase pengembangan dan tonggak untuk proses modernisasi, dan rencana aksi untuk mengatasi kesenjangan yang diidentifikasi. Untuk informasi selengkapnya, lihat [Mengevaluasi kesiapan modernisasi untuk aplikasi](#) di Cloud. AWS

aplikasi monolitik (monolit)

Aplikasi yang berjalan sebagai layanan tunggal dengan proses yang digabungkan secara ketat. Aplikasi monolitik memiliki beberapa kelemahan. Jika satu fitur aplikasi mengalami lonjakan permintaan, seluruh arsitektur harus diskalakan. Menambahkan atau meningkatkan fitur aplikasi monolitik juga menjadi lebih kompleks ketika basis kode tumbuh. Untuk mengatasi masalah ini, Anda dapat menggunakan arsitektur microservices. Untuk informasi lebih lanjut, lihat [Menguraikan monolit](#) menjadi layanan mikro.

klasifikasi multiclass

Sebuah proses yang membantu menghasilkan prediksi untuk beberapa kelas (memprediksi satu dari lebih dari dua hasil). Misalnya, model ML mungkin bertanya “Apakah produk ini buku, mobil, atau telepon?” atau “Kategori produk mana yang paling menarik bagi pelanggan ini?”

infrastruktur yang bisa berubah

Model yang memperbarui dan memodifikasi infrastruktur yang ada untuk beban kerja produksi. Untuk meningkatkan konsistensi, keandalan, dan prediktabilitas, AWS Well-Architected Framework merekomendasikan penggunaan infrastruktur yang [tidak](#) dapat diubah sebagai praktik terbaik.

O

OAC

Lihat [kontrol akses asal](#).

OAI

Lihat [identitas akses asal](#).

OCM

Lihat [manajemen perubahan organisasi](#).

migrasi offline

Metode migrasi di mana beban kerja sumber diturunkan selama proses migrasi. Metode ini melibatkan waktu henti yang diperpanjang dan biasanya digunakan untuk beban kerja kecil dan tidak kritis.

OI

Lihat [integrasi operasi](#).

OLA

Lihat [perjanjian tingkat operasional](#).

migrasi online

Metode migrasi di mana beban kerja sumber disalin ke sistem target tanpa diambil offline. Aplikasi yang terhubung ke beban kerja dapat terus berfungsi selama migrasi. Metode ini melibatkan waktu henti nol hingga minimal dan biasanya digunakan untuk beban kerja produksi yang kritis.

perjanjian tingkat operasional (OLA)

Perjanjian yang menjelaskan apa yang dijanjikan kelompok TI fungsional untuk diberikan satu sama lain, untuk mendukung perjanjian tingkat layanan (SLA).

Tinjauan Kesiapan Operasional (ORR)

Daftar pertanyaan dan praktik terbaik terkait yang membantu Anda memahami, mengevaluasi, mencegah, atau mengurangi ruang lingkup insiden dan kemungkinan kegagalan. Untuk informasi lebih lanjut, lihat [Ulasan Kesiapan Operasional \(ORR\)](#) dalam Kerangka Kerja Well-ArchitectedAWS.

integrasi operasi (OI)

Proses modernisasi operasi di cloud, yang melibatkan perencanaan kesiapan, otomatisasi, dan integrasi. Untuk informasi selengkapnya, lihat [panduan integrasi operasi](#).

jejak organisasi

Jejak yang dibuat oleh AWS CloudTrail itu mencatat semua peristiwa untuk semua Akun AWS dalam organisasi diAWS Organizations. Jejak ini dibuat di setiap Akun AWS bagian organisasi dan melacak aktivitas di setiap akun. Untuk informasi selengkapnya, lihat [Membuat jejak untuk organisasi](#) dalam CloudTrail dokumentasi.

manajemen perubahan organisasi (OCM)

Kerangka kerja untuk mengelola transformasi bisnis utama yang mengganggu dari perspektif orang, budaya, dan kepemimpinan. OCM membantu organisasi mempersiapkan, dan transisi ke, sistem dan strategi baru dengan mempercepat adopsi perubahan, mengatasi masalah transisi, dan mendorong perubahan budaya dan organisasi. Dalam strategi AWS migrasi, kerangka kerja ini disebut percepatan orang, karena kecepatan perubahan yang diperlukan dalam proyek adopsi cloud. Untuk informasi lebih lanjut, lihat [panduan OCM](#).

kontrol akses asal (OAC)

Di CloudFront, opsi yang disempurnakan untuk membatasi akses untuk mengamankan konten Amazon Simple Storage Service (Amazon S3) Anda. OAC mendukung semua bucket S3 di semua Wilayah AWS, enkripsi sisi server dengan AWS KMS (SSE-KMS), dan dinamis dan permintaan ke bucket S3. PUT DELETE

identitas akses asal (OAI)

Di CloudFront, opsi untuk membatasi akses untuk mengamankan konten Amazon S3 Anda. Saat Anda menggunakan OAI, CloudFront buat prinsipal yang dapat diautentikasi oleh Amazon S3. Prinsipal yang diautentikasi dapat mengakses konten dalam bucket S3 hanya melalui distribusi tertentu. CloudFront Lihat juga [OAC](#), yang menyediakan kontrol akses yang lebih terperinci dan ditingkatkan.

ORR

Lihat [tinjauan kesiapan operasional](#).

keluar (jalan keluar) VPC

Dalam arsitektur AWS multi-akun, VPC yang menangani koneksi jaringan yang dimulai dari dalam aplikasi. [Arsitektur Referensi AWS Keamanan](#) merekomendasikan pengaturan akun Jaringan Anda dengan VPC masuk, keluar, dan inspeksi untuk melindungi antarmuka dua arah antara aplikasi Anda dan internet yang lebih luas.

P

batas izin

Kebijakan manajemen IAM yang dilampirkan pada prinsipal IAM untuk menetapkan izin maksimum yang dapat dimiliki pengguna atau peran. Untuk informasi selengkapnya, lihat [Batas izin](#) dalam dokumentasi IAM.

Informasi Identifikasi Pribadi (PII)

Informasi yang, jika dilihat secara langsung atau dipasangkan dengan data terkait lainnya, dapat digunakan untuk menyimpulkan identitas individu secara wajar. Contoh PII termasuk nama, alamat, dan informasi kontak.

PII

Lihat informasi yang [dapat diidentifikasi secara pribadi](#).

buku pedoman

Serangkaian langkah yang telah ditentukan sebelumnya yang menangkap pekerjaan yang terkait dengan migrasi, seperti mengirimkan fungsi operasi inti di cloud. Buku pedoman dapat berupa skrip, runbook otomatis, atau ringkasan proses atau langkah-langkah yang diperlukan untuk mengoperasikan lingkungan modern Anda.

kebijakan

[Objek yang dapat menentukan izin \(lihat kebijakan berbasis identitas\), menentukan kondisi akses \(lihat kebijakan berbasis sumber daya\), atau menentukan izin maksimum untuk semua akun dalam organisasi di \(lihat kebijakan kontrol layanan\). AWS Organizations](#)

ketekunan poliglott

Secara independen memilih teknologi penyimpanan data microservice berdasarkan pola akses data dan persyaratan lainnya. Jika layanan mikro Anda memiliki teknologi penyimpanan data yang sama, mereka dapat menghadapi tantangan implementasi atau mengalami kinerja yang buruk. Layanan mikro lebih mudah diimplementasikan dan mencapai kinerja dan skalabilitas yang lebih baik jika mereka menggunakan penyimpanan data yang paling sesuai dengan kebutuhan mereka. Untuk informasi selengkapnya, lihat [Mengaktifkan persistensi data di layanan mikro](#).

penilaian portofolio

Proses menemukan, menganalisis, dan memprioritaskan portofolio aplikasi untuk merencanakan migrasi. Untuk informasi selengkapnya, lihat [Mengevaluasi kesiapan migrasi](#).

predikat

Kondisi kueri yang mengembalikan `true` atau `false`, biasanya terletak di `WHERE` klausa.

predikat pushdown

Teknik optimasi kueri database yang menyaring data dalam kueri sebelum transfer. Ini mengurangi jumlah data yang harus diambil dan diproses dari database relasional, dan meningkatkan kinerja kueri.

kontrol preventif

Kontrol keamanan yang dirancang untuk mencegah suatu peristiwa terjadi. Kontrol ini adalah garis pertahanan pertama untuk membantu mencegah akses tidak sah atau perubahan yang tidak diinginkan ke jaringan Anda. Untuk informasi selengkapnya, lihat [Kontrol pencegahan dalam Menerapkan kontrol](#) keamanan pada AWS.

principal

Sebuah entitas dalam AWS yang dapat melakukan tindakan dan mengakses sumber daya. Entitas ini biasanya merupakan pengguna root untuk Akun AWS, peran IAM, atau pengguna. Untuk informasi selengkapnya, lihat Prinsip dalam [istilah dan konsep Peran](#) dalam dokumentasi IAM.

Privasi oleh Desain

Pendekatan dalam rekayasa sistem yang memperhitungkan privasi di seluruh proses rekayasa. zona yang dihosting pribadi

Container yang menyimpan informasi tentang bagaimana Anda ingin Amazon Route 53 merespons kueri DNS untuk domain dan subdomainnya dalam satu atau beberapa VPC. Untuk informasi selengkapnya, lihat [Bekerja dengan zona yang dihosting pribadi](#) di dokumentasi Route 53.

kontrol proaktif

[Kontrol keamanan](#) yang dirancang untuk mencegah penyebaran sumber daya yang tidak sesuai. Kontrol ini memindai sumber daya sebelum disediakan. Jika sumber daya tidak sesuai dengan kontrol, maka itu tidak disediakan. Untuk informasi selengkapnya, lihat [panduan referensi Kontrol](#) dalam AWS Control Tower dokumentasi dan lihat [Kontrol proaktif](#) dalam Menerapkan kontrol keamanan pada AWS.

lingkungan produksi

Lihat [lingkungan](#).

pseudonimisasi

Proses penggantian pengenalan pribadi dalam kumpulan data dengan nilai placeholder. Pseudonimisasi dapat membantu melindungi privasi pribadi. Data pseudonim masih dianggap sebagai data pribadi.

Q

rencana kueri

Serangkaian langkah, seperti instruksi, yang digunakan untuk mengakses data dalam sistem database relasional SQL.

regresi rencana kueri

Ketika pengoptimal layanan database memilih rencana yang kurang optimal daripada sebelum perubahan yang diberikan ke lingkungan database. Hal ini dapat disebabkan oleh perubahan statistik, kendala, pengaturan lingkungan, pengikatan parameter kueri, dan pembaruan ke mesin database.

R

Matriks RACI

Lihat [bertanggung jawab, akuntabel, dikonsultasikan, diinformasikan \(RACI\)](#).

ransomware

Perangkat lunak berbahaya yang dirancang untuk memblokir akses ke sistem komputer atau data sampai pembayaran dilakukan.

Matriks RASCI

Lihat [bertanggung jawab, akuntabel, dikonsultasikan, diinformasikan \(RACI\)](#).

RCAC

Lihat [kontrol akses baris dan kolom](#).

replika baca

Salinan database yang digunakan untuk tujuan read-only. Anda dapat merutekan kueri ke replika baca untuk mengurangi beban pada database utama Anda.

arsitek ulang

Lihat [7 Rs](#).

tujuan titik pemulihan (RPO)

Jumlah waktu maksimum yang dapat diterima sejak titik pemulihan data terakhir. Ini menentukan apa yang dianggap sebagai hilangnya data yang dapat diterima antara titik pemulihan terakhir dan gangguan layanan.

tujuan waktu pemulihan (RTO)

Penundaan maksimum yang dapat diterima antara gangguan layanan dan pemulihan layanan.

refactor

Lihat [7 Rs](#).

Wilayah

Kumpulan AWS sumber daya di wilayah geografis. Masing-masing Wilayah AWS terisolasi dan independen dari yang lain untuk memberikan toleransi kesalahan, stabilitas, dan ketahanan. Untuk informasi selengkapnya, lihat [Mengelola Wilayah AWS](#) di Referensi Umum AWS.

regresi

Teknik ML yang memprediksi nilai numerik. Misalnya, untuk memecahkan masalah “Berapa harga rumah ini akan dijual?” Model ML dapat menggunakan model regresi linier untuk memprediksi harga jual rumah berdasarkan fakta yang diketahui tentang rumah (misalnya, luas persegi).

rehost

Lihat [7 Rs](#).

melepaskan

Dalam proses penyebaran, tindakan mempromosikan perubahan pada lingkungan produksi.

memindahkan

Lihat [7 Rs](#).

memplatform ulang

Lihat [7 Rs](#).

pembelian kembali

Lihat [7 Rs](#).

kebijakan berbasis sumber daya

Kebijakan yang dilampirkan ke sumber daya, seperti bucket Amazon S3, titik akhir, atau kunci enkripsi. Jenis kebijakan ini menentukan prinsipal mana yang diizinkan mengakses, tindakan yang didukung, dan kondisi lain yang harus dipenuhi.

matriks yang bertanggung jawab, akuntabel, dikonsultasikan, diinformasikan (RACI)

Matriks yang mendefinisikan peran dan tanggung jawab untuk semua pihak yang terlibat dalam kegiatan migrasi dan operasi cloud. Nama matriks berasal dari jenis tanggung jawab yang didefinisikan dalam matriks: bertanggung jawab (R), akuntabel (A), dikonsultasikan (C), dan diinformasikan (I). Tipe dukungan (S) adalah opsional. Jika Anda menyertakan dukungan, matriks disebut matriks RASCI, dan jika Anda mengecualikannya, itu disebut matriks RACI.

kontrol responsif

Kontrol keamanan yang dirancang untuk mendorong remediasi efek samping atau penyimpangan dari garis dasar keamanan Anda. Untuk informasi selengkapnya, lihat [Kontrol responsif](#) dalam Menerapkan kontrol keamanan pada AWS.

melestarikan

Lihat [7 Rs](#).

pensiun

Lihat [7 Rs](#).

rotasi

Proses memperbarui [rahasia](#) secara berkala untuk membuatnya lebih sulit bagi penyerang untuk mengakses kredensial.

kontrol akses baris dan kolom (RCAC)

Penggunaan ekspresi SQL dasar dan fleksibel yang telah menetapkan aturan akses. RCAC terdiri dari izin baris dan topeng kolom.

RPO

Lihat [tujuan titik pemulihan](#).

RTO

Lihat [tujuan waktu pemulihan](#).

buku runbook

Satu set prosedur manual atau otomatis yang diperlukan untuk melakukan tugas tertentu. Ini biasanya dibangun untuk merampingkan operasi berulang atau prosedur dengan tingkat kesalahan yang tinggi.

D

SAML 2.0

Standar terbuka yang digunakan oleh banyak penyedia identitas (IdPs). Fitur ini memungkinkan sistem masuk tunggal gabungan (SSO), sehingga pengguna dapat masuk ke AWS Management Console atau memanggil operasi AWS API tanpa Anda harus membuat pengguna di IAM untuk semua orang di organisasi Anda. Untuk informasi lebih lanjut tentang federasi berbasis SAMP 2.0, lihat [Tentang federasi berbasis SAMP 2.0](#) dalam dokumentasi IAM.

SCP

Lihat [kebijakan kontrol layanan](#).

Rahasia

Dalam AWS Secrets Manager, informasi rahasia atau terbatas, seperti kata sandi atau kredensi pengguna, yang Anda simpan dalam bentuk terenkripsi. Ini terdiri dari nilai rahasia dan metadatanya. Nilai rahasia dapat berupa biner, string tunggal, atau beberapa string. Untuk informasi selengkapnya, lihat [Rahasia](#) dalam dokumentasi Secrets Manager.

kontrol keamanan

Pagar pembatas teknis atau administratif yang mencegah, mendeteksi, atau mengurangi kemampuan pelaku ancaman untuk mengeksploitasi kerentanan keamanan. [Ada empat jenis kontrol keamanan utama: preventif, detektif, responsif, dan proaktif.](#)

pengerasan keamanan

Proses mengurangi permukaan serangan untuk membuatnya lebih tahan terhadap serangan. Ini dapat mencakup tindakan seperti menghapus sumber daya yang tidak lagi diperlukan, menerapkan praktik keamanan terbaik untuk memberikan hak istimewa paling sedikit, atau menonaktifkan fitur yang tidak perlu dalam file konfigurasi.

sistem informasi keamanan dan manajemen acara (SIEM)

Alat dan layanan yang menggabungkan sistem manajemen informasi keamanan (SIM) dan manajemen acara keamanan (SEM). Sistem SIEM mengumpulkan, memantau, dan menganalisis data dari server, jaringan, perangkat, dan sumber lain untuk mendeteksi ancaman dan pelanggaran keamanan, dan untuk menghasilkan peringatan.

otomatisasi respons keamanan

Tindakan yang telah ditentukan dan diprogram yang dirancang untuk secara otomatis merespons atau memulihkan peristiwa keamanan. Otomatisasi ini berfungsi sebagai kontrol keamanan [detektif](#) atau [responsif](#) yang membantu Anda menerapkan praktik terbaik AWS keamanan. Contoh tindakan respons otomatis termasuk memodifikasi grup keamanan VPC, menambal instans Amazon EC2, atau memutar kredensial.

enkripsi sisi server

Enkripsi data di tujuannya, oleh Layanan AWS yang menerimanya.

kebijakan kontrol layanan (SCP)

Kebijakan yang menyediakan kontrol terpusat atas izin untuk semua akun di organisasi. AWS Organizations SCP menentukan pagar pembatas atau menetapkan batasan pada tindakan yang dapat didelegasikan oleh administrator kepada pengguna atau peran. Anda dapat menggunakan SCP sebagai daftar izin atau daftar penolakan, untuk menentukan layanan atau tindakan mana yang diizinkan atau dilarang. Untuk informasi selengkapnya, lihat [Kebijakan kontrol layanan](#) dalam AWS Organizations dokumentasi.

titik akhir layanan

URL titik masuk untuk fileLayanan AWS. Anda dapat menggunakan endpoint untuk terhubung secara terprogram ke layanan target. Untuk informasi selengkapnya, lihat [Layanan AWStitik akhir](#) di Referensi Umum AWS.

perjanjian tingkat layanan (SLA)

Perjanjian yang menjelaskan apa yang dijanjikan tim TI untuk diberikan kepada pelanggan mereka, seperti waktu kerja dan kinerja layanan.

indikator tingkat layanan (SLI)

Pengukuran aspek kinerja layanan, seperti tingkat kesalahan, ketersediaan, atau throughputnya.

tujuan tingkat layanan (SLO)

Metrik target yang mewakili kesehatan layanan, yang diukur dengan indikator [tingkat layanan](#).

model tanggung jawab bersama

Model yang menjelaskan tanggung jawab yang Anda bagikan AWS untuk keamanan dan kepatuhan cloud. AWS bertanggung jawab atas keamanan cloud, sedangkan Anda bertanggung jawab atas keamanan di cloud. Untuk informasi selengkapnya, lihat [Model tanggung jawab bersama](#).

SIEM

Lihat [informasi keamanan dan sistem manajemen acara](#).

titik kegagalan tunggal (SPOF)

Kegagalan dalam satu komponen penting dari aplikasi yang dapat mengganggu sistem.

SLA

Lihat [perjanjian tingkat layanan](#).

SLI

Lihat [indikator tingkat layanan](#).

SLO

Lihat [tujuan tingkat layanan](#).

split-and-seed model

Pola untuk menskalakan dan mempercepat proyek modernisasi. Ketika fitur baru dan rilis produk didefinisikan, tim inti berpisah untuk membuat tim produk baru. Ini membantu meningkatkan kemampuan dan layanan organisasi Anda, meningkatkan produktivitas pengembang, dan mendukung inovasi yang cepat. Untuk informasi lebih lanjut, lihat [Pendekatan bertahap untuk memodernisasi aplikasi](#) di AWS Cloud

SPOF

Lihat [satu titik kegagalan](#).

skema bintang

Struktur organisasi database yang menggunakan satu tabel fakta besar untuk menyimpan data transaksional atau terukur dan menggunakan satu atau lebih tabel dimensi yang lebih kecil untuk menyimpan atribut data. Struktur ini dirancang untuk digunakan dalam [gudang data](#) atau untuk tujuan intelijen bisnis.

pola ara pencekik

Pendekatan untuk memodernisasi sistem monolitik dengan menulis ulang secara bertahap dan mengganti fungsionalitas sistem sampai sistem warisan dapat dinonaktifkan. Pola ini menggunakan analogi pohon ara yang tumbuh menjadi pohon yang sudah mapan dan akhirnya mengatasi dan menggantikan inangnya. Pola ini [diperkenalkan oleh Martin Fowler](#) sebagai cara untuk mengelola risiko saat menulis ulang sistem monolitik. Untuk contoh cara menerapkan pola ini, lihat [Memodernisasi layanan web Microsoft ASP.NET \(ASMX\) lama secara bertahap menggunakan container dan Amazon API Gateway](#).

subnet

Rentang alamat IP dalam VPC Anda. Subnet harus berada di Availability Zone tunggal.

enkripsi simetris

Algoritma enkripsi yang menggunakan kunci yang sama untuk mengenkripsi dan mendekripsi data.

pengujian sintetis

Menguji sistem dengan cara yang mensimulasikan interaksi pengguna untuk mendeteksi potensi masalah atau untuk memantau kinerja. Anda dapat menggunakan [Amazon CloudWatch Synthetics](#) untuk membuat tes ini.

T

tag

Pasangan nilai kunci yang bertindak sebagai metadata untuk mengatur sumber daya Anda. AWS Tanda dapat membantu Anda mengelola, mengidentifikasi, mengatur, dan memfilter sumber daya. Untuk informasi selengkapnya, lihat [Menandai AWS sumber daya Anda](#).

variabel target

Nilai yang Anda coba prediksi dalam ML yang diawasi. Ini juga disebut sebagai variabel hasil. Misalnya, dalam pengaturan manufaktur, variabel target bisa menjadi cacat produk.

daftar tugas

Alat yang digunakan untuk melacak kemajuan melalui runbook. Daftar tugas berisi ikhtisar runbook dan daftar tugas umum yang harus diselesaikan. Untuk setiap tugas umum, itu termasuk perkiraan jumlah waktu yang dibutuhkan, pemilik, dan kemajuan.

lingkungan uji

Lihat [lingkungan](#).

pelatihan

Untuk menyediakan data bagi model ML Anda untuk dipelajari. Data pelatihan harus berisi jawaban yang benar. Algoritma pembelajaran menemukan pola dalam data pelatihan yang memetakan atribut data input ke target (jawaban yang ingin Anda prediksi). Ini menghasilkan model ML yang menangkap pola-pola ini. Anda kemudian dapat menggunakan model ML untuk membuat prediksi pada data baru yang Anda tidak tahu targetnya.

gerbang transit

Hub transit jaringan yang dapat Anda gunakan untuk menghubungkan VPC dan jaringan lokal Anda. Untuk informasi selengkapnya, lihat [Apa itu gateway transit](#) dalam AWS Transit Gateway dokumentasi.

alur kerja berbasis batang

Pendekatan di mana pengembang membangun dan menguji fitur secara lokal di cabang fitur dan kemudian menggabungkan perubahan tersebut ke cabang utama. Cabang utama kemudian dibangun untuk pengembangan, praproduksi, dan lingkungan produksi, secara berurutan.

akses tepercaya

Memberikan izin ke layanan yang Anda tentukan untuk melakukan tugas di organisasi Anda di dalam AWS Organizations dan di akunnya atas nama Anda. Layanan tepercaya menciptakan peran terkait layanan di setiap akun, ketika peran itu diperlukan, untuk melakukan tugas manajemen untuk Anda. Untuk informasi selengkapnya, lihat [Menggunakan AWS Organizations dengan AWS layanan lain](#) dalam AWS Organizations dokumentasi.

penyetelan

Untuk mengubah aspek proses pelatihan Anda untuk meningkatkan akurasi model ML. Misalnya, Anda dapat melatih model ML dengan membuat set pelabelan, menambahkan label, dan kemudian mengulangi langkah-langkah ini beberapa kali di bawah pengaturan yang berbeda untuk mengoptimalkan model.

tim dua pizza

Sebuah DevOps tim kecil yang bisa Anda beri makan dengan dua pizza. Ukuran tim dua pizza memastikan peluang terbaik untuk berkolaborasi dalam pengembangan perangkat lunak.

U

waswas

Sebuah konsep yang mengacu pada informasi yang tidak tepat, tidak lengkap, atau tidak diketahui yang dapat merusak keandalan model ML prediktif. Ada dua jenis ketidakpastian: ketidakpastian epistemik disebabkan oleh data yang terbatas dan tidak lengkap, sedangkan ketidakpastian aleatorik disebabkan oleh kebisingan dan keacakan yang melekat dalam data. Untuk informasi lebih lanjut, lihat panduan [Mengukur ketidakpastian dalam sistem pembelajaran mendalam](#).

tugas yang tidak terdiferensiasi

Juga dikenal sebagai angkat berat, pekerjaan yang diperlukan untuk membuat dan mengoperasikan aplikasi tetapi itu tidak memberikan nilai langsung kepada pengguna akhir atau memberikan keunggulan kompetitif. Contoh tugas yang tidak terdiferensiasi termasuk pengadaan, pemeliharaan, dan perencanaan kapasitas.

lingkungan atas

Lihat [lingkungan](#).

V

menyedot debu

Operasi pemeliharaan database yang melibatkan pembersihan setelah pembaruan tambahan untuk merebut kembali penyimpanan dan meningkatkan kinerja.

kendali versi

Proses dan alat yang melacak perubahan, seperti perubahan kode sumber dalam repositori.

Peering VPC

Koneksi antara dua VPC yang memungkinkan Anda merutekan lalu lintas dengan menggunakan alamat IP pribadi. Untuk informasi selengkapnya, lihat [Apa itu peering VPC](#) di dokumentasi VPC Amazon.

kerentanan

Kelemahan perangkat lunak atau perangkat keras yang membahayakan keamanan sistem.

W

cache hangat

Cache buffer yang berisi data saat ini dan relevan yang sering diakses. Instance database dapat membaca dari cache buffer, yang lebih cepat daripada membaca dari memori utama atau disk.

data hangat

Data yang jarang diakses. Saat menanyakan jenis data ini, kueri yang cukup lambat biasanya dapat diterima.

fungsi jendela

Fungsi SQL yang melakukan perhitungan pada sekelompok baris yang berhubungan dengan catatan saat ini. Fungsi jendela berguna untuk memproses tugas, seperti menghitung rata-rata bergerak atau mengakses nilai baris berdasarkan posisi relatif dari baris saat ini.

beban kerja

Kumpulan sumber daya dan kode yang memberikan nilai bisnis, seperti aplikasi yang dihadapi pelanggan atau proses backend.

aliran kerja

Grup fungsional dalam proyek migrasi yang bertanggung jawab atas serangkaian tugas tertentu. Setiap alur kerja independen tetapi mendukung alur kerja lain dalam proyek. Misalnya, alur kerja portofolio bertanggung jawab untuk memprioritaskan aplikasi, perencanaan gelombang, dan mengumpulkan metadata migrasi. Alur kerja portofolio mengirimkan aset ini ke alur kerja migrasi, yang kemudian memigrasikan server dan aplikasi.

CACING

Lihat [menulis sekali, baca banyak](#).

WQF

Lihat [Kerangka Kualifikasi Beban Kerja AWS](#).

tulis sekali, baca banyak (WORM)

Model penyimpanan yang menulis data satu kali dan mencegah data dihapus atau dimodifikasi. Pengguna yang berwenang dapat membaca data sebanyak yang diperlukan, tetapi mereka tidak dapat mengubahnya. Infrastruktur penyimpanan data ini dianggap [tidak dapat diubah](#).

Z

eksploitasi zero-day

Serangan, biasanya malware, yang memanfaatkan kerentanan [zero-day](#).

kerentanan zero-day

Cacat atau kerentanan yang tak tanggung-tanggung dalam sistem produksi. Aktor ancaman dapat menggunakan jenis kerentanan ini untuk menyerang sistem. Pengembang sering menyadari kerentanan sebagai akibat dari serangan tersebut.

aplikasi zombie

Aplikasi yang memiliki CPU rata-rata dan penggunaan memori di bawah 5 persen. Dalam proyek migrasi, adalah umum untuk menghentikan aplikasi ini.

Terjemahan disediakan oleh mesin penerjemah. Jika konten terjemahan yang diberikan bertentangan dengan versi bahasa Inggris aslinya, utamakan versi bahasa Inggris.